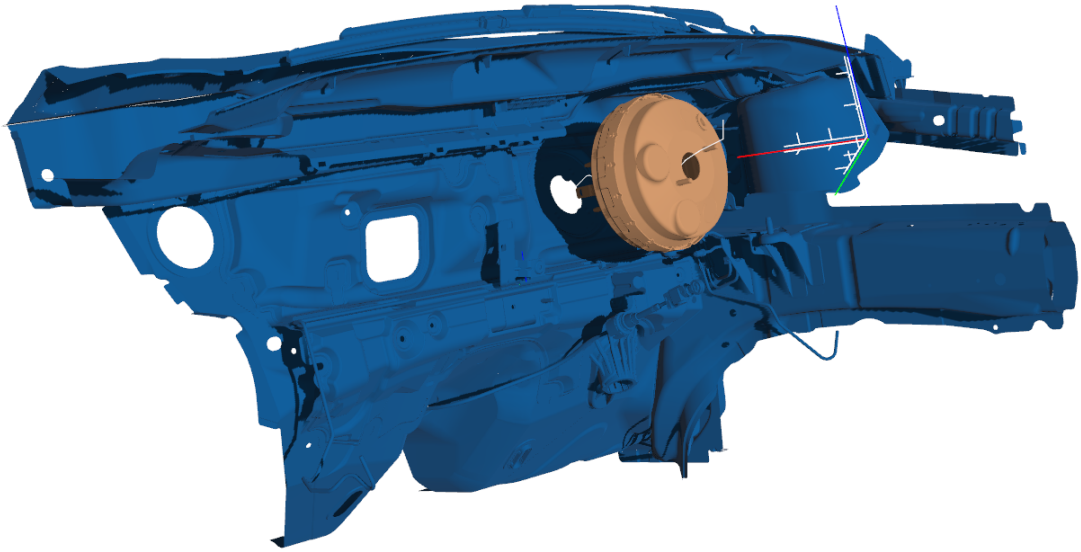


Enhancing Rigid Body Path Planning via Anchor Frame Optimization

Anchor Frame Impact Assessment and Strategization

Master's thesis in Engineering Mathematics and Computational Science

SAMUEL ALVINGE, AXEL BLOM

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Master's thesis 2024

Enhancing Rigid Body Path Planning via Anchor Frame Optimization

Anchor Frame Impact Assessment and Strategization

SAMUEL ALVINGE,
AXEL BLOM

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
Gothenburg, Sweden 2024

Enhancing Rigid Body Path Planning via Anchor Frame Optimization
Anchor Frame Impact Assessment and Strategization
SAMUEL ALVINGE, AXEL BLOM

Supervisors: Robert Bohlin and Edvin Åblad, Fraunhofer Chalmers Centre
Examiner: Ann-Brith Strömberg, Mathematical Sciences

Master's Thesis 2024
Department of Mathematical Sciences
Division of Applied Mathematics and Statistics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Path planning problem for a brake booster. Model courtesy of Volvo Cars.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Enhancing Rigid Body Path Planning via Anchor Frame Optimization
Anchor Frame Impact Assessment and Strategization
SAMUEL ALVINGE, AXEL BLOM
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

In the field of computational geometry, a classical problem is the piano movers' problem, which involves finding a path for a rigid body, such as a lumbering piano, through a confined and challenging environment. Solving this requires path planning algorithms that precisely control and manipulate the object's position and orientation. Most path planning algorithms depend on a so-called anchor frame, which is fixed to the rigid body. Many path planning algorithms find a path through a six-dimensional configuration space rather than the physical space, often from the perspective of the anchor frame. This study assesses the impact of the anchor frame's pose on the performance of Industrial Path Solutions' path planning algorithm. Strategies are proposed that optimize the anchor frame's orientation and position for enhanced performance. Sampled measurements from the path planner are utilized by the strategies to predict the subsequent movement of the rigid body. Results indicate that appropriate alignment of the anchor frame significantly improves the navigability of the configuration space, reducing computational efforts and enhancing overall performance. However, the placement strategy remains challenging due to the local nature of the available data, and no successful strategy has been found. The findings underscore the potential of refined orientation strategies and highlight the need for further research into robust placement techniques.

Keywords: path planning, rigid body, optimization, efficiency, geometry.

Acknowledgements

We would like to express our sincere gratitude to our supervisors, Robert Bohlin and Edvin Åblad, for their exceptional guidance throughout this project. Your mathematical insights and expertise have been invaluable, and your positive support has significantly boosted our morale. We would also like to extend our heartfelt thanks to Ann-Brith Strömberg for generously taking the time to serve as our examiner.

Samuel Alvinge, Axel Blom, Gothenburg, June 2024

Nomenclature

Indices

i, j, k	Indices for Elements in Sets or Vectors

Sets

$SO(3)$	Set of Rotations in Three Dimensions
$SE(3)$	Set of Rigid Body Transformations in Three Dimensions
$\mathcal{C}$	Configuration Space
$\mathbb{R}^3$	Three-Dimensional Euclidean Space
$\mathcal{S}_\xi$	Candidate Twist Set

Parameters

θ, τ	Time Parameters
Σ	Covariance Matrix
σ	Standard Deviation
α	Search Cone Angle
n	Number of Points on Sphere
l	Anchor Frame Displacement Limit

Variables

φ_z	Rotation About z-Axis
$\boldsymbol{p}, \boldsymbol{q}$	Points in Global Coordinates
$\tilde{\boldsymbol{p}}, \tilde{\boldsymbol{q}}$	Points in Local Coordinates
$\dot{\boldsymbol{p}}$	Time Derivative of a Point $\boldsymbol{p}$
$\boldsymbol{a}, \boldsymbol{b}$	3D Vectors
$\bar{\boldsymbol{a}}, \bar{\boldsymbol{b}}$	Homogeneous Vectors

$\boldsymbol{\xi}$	Exponential Coordinates (Parameterized Twist)
$\bar{\xi}$	Homogeneous Twist (Matrix Form)
$\boldsymbol{\omega}$	Angular Velocity
$\hat{\omega}$	Angular Velocity (Cross Matrix)
$\boldsymbol{v}$	Translational Velocity
$\hat{\boldsymbol{d}}$	Local Translational Prediction
$\boldsymbol{d}$	Adjusted Translational Prediction
$\boldsymbol{\varepsilon}$	Noise Vector
ζ	Configuration

Objects

B	Rigid Body
A_B, A	Rigid Body Anchor Frame
R	Rotation Matrix
$\mathbb{I}$	Identity Matrix
G	Global Coordinate Frame
S	Static Geometry
$\mathbf{0}$	Zero Vector

Functions

g	Rigid Body Transformation
T	Homogeneous Transformation
h	Displacement Evaluation Function
f	Optimization Objective Function

Distributions

$\mathcal{N}$	Gaussian Distribution

Contents

List of Figures

List of Tables

1
Introduction

The introduction chapter provides an overview of the report's objectives and context. An introduction to path planning is given, along with a description of how the project aims to enhance the efficiency of an existing path planning algorithm. The chapter details the significance of the placement and orientation of the so-called anchor frame in the path planning algorithm, offering insights into the available tools and techniques. Additionally, the aim, scope, and limitations of the report are established.

The project has been carried out on behalf of Fraunhofer Chalmers Centre (FCC), a research institute in industrial mathematics [1]. FCC offers contract research, services, algorithms, and software based on advanced mathematics in Modeling, Simulation, and Optimization. FCC is funded in three ways by approximately equal parts: base funding, industry projects, and grants. Fraunhofer, a world-leading applied research organization [2], along with Chalmers, provides equal contributions towards the base funding of FCC [3].

Industrial Path Solutions (IPS) [4] is a spin-off company from FCC. As a result of successful research during the first few years of FCC, there was a demand for a commercial product to house the inventions produced by FCC. All tools and products developed by IPS are implemented in a mathematics-based software suite, hereby referred to as the IPS software suite. The IPS software suite is mathematics-based software that allows engineers to import models from computer-aided design (CAD) systems and simulate various complex manufacturing processes.

1.1 Background

Automatic path planning has been a significant research topic for over half a century [5] and continues to expand today [6], particularly in fields such as robotics [7, 8, 9], autonomous vehicles [10], and manufacturing [11]. One type of path planning is that of rigid bodies, which the following example introduces: You are moving to a new house and face the challenging task of moving your couch through the door frame, depicted in Figure 1.1. The situation becomes even more complicated when you realize there are several obstacles in the way. Moving the couch by hand would be cumbersome and would require very precise movements.

Luckily, you have an exact 3D model of your couch, the door frame, and all obstacles.

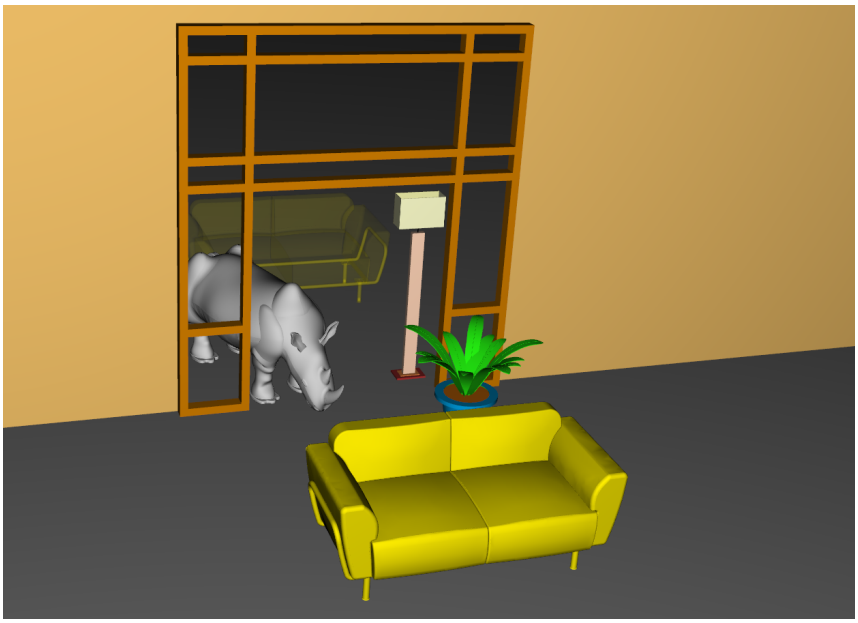


Figure 1.1: A problem of finding a path for the couch through the door frame without colliding with the plant, lamp, or rhinoceros.

This allows you to use a path planning algorithm to determine if there exists a collision-free path through the door frame. The solution to the problem, found by the path planning algorithm, is illustrated in Figure 1.2.

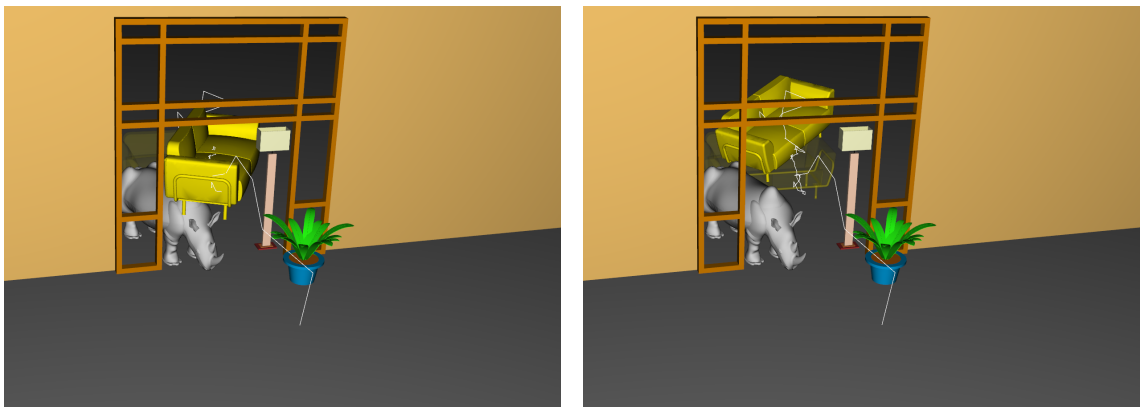


Figure 1.2: A couch, moving through the door frame along a path found by a path planning algorithm. The white line represents translational movements the couch makes to reach its goal position.

Path planning is more commonly used in industrial settings, particularly to virtually verify that assembly processes — involving precise combination, fitting, and alignment of individual components to construct a final product — are feasible. Users can iteratively add new parts to a model while ensuring that each part can still be installed and removed. Often, users of a path planning algorithm are more interested in whether a collision-free path exists than in the specific details of the path returned. By enabling virtual verification, path planning reduces the need for

physical prototypes which helps manufacturers save time, costs, and materials while ensuring precision and reliability in product assembly and disassembly. This not only benefits companies directly but also supports their efforts to meet the growing global demand for resource-efficient manufacturing [12].

One of the earlier products developed at Fraunhofer Chalmers Centre (FCC) is a rigid body path planning algorithm, which is part of the IPS software suite, hereby referred to as the IPS path planning algorithm. Specifically, the algorithm allows users to define start and goal configurations for the rigid body and obtain a collision-free path through surrounding obstacles, if such a path exists.

Path planning is generally conducted in an abstract space known as the configuration space, rather than in the physical space of the rigid body. This space represents all possible configurations of a physical object, such as the angles of a robot arm or the pose of a rigid body, where a pose combines position and orientation. Configurations in this space are evaluated for collisions, and the collision-free subset is particularly important. Path planning involves finding a path through this subset from the start to the end configurations, resulting in a collision-free path.

The IPS path planning algorithm differs from traditional graph-based pathfinding algorithms. Path planning should not be confused with well-known graph-based pathfinding algorithms, such as Dijkstra's algorithm [5] and the A* algorithm [13]. These algorithms are designed to find the shortest path in explicit weighted graphs, whereas rigid body path planning focuses on building the graph itself. Theoretically, it would be possible to evaluate every point in the configuration space within a grid and then use a path finding algorithm to find a path between the start and goal configurations. This would, however, require an impractical amount of computationally expensive collision checks, due to the high dimensionality of the configuration space. Instead, the IPS software suite uses a search algorithm that progressively explores the configuration space. During the search, the algorithm samples configurations and calculates the clearance between the rigid body and surrounding static geometry. Static geometry refers to rigid, fixed obstacles, as is the case for all obstacles in this project. The algorithm is bidirectional, searching from both the start and goal configurations in favorable directions until the paths converge into a connected path.

Calculating clearance is a crucial and computationally demanding task in the path planning algorithm, which refers to the smallest distance between object models. The models are commonly composed of thousands, sometimes millions, of mesh triangles. Consequently, repeatedly computing the smallest distance between models is the most computationally expensive part of the algorithm. This step has to be performed for each sample the path planner makes.

To introduce the concept of anchor frames, let us continue with the previous example. Imagine that you have two different options for controlling the motion of the couch, as illustrated by two hands in Figure 1.3. One hand represents moving the couch from its side, perhaps by physically grabbing it. The other hand represents

controlling the couch from a distance, by some imaginary force. In this scenario,

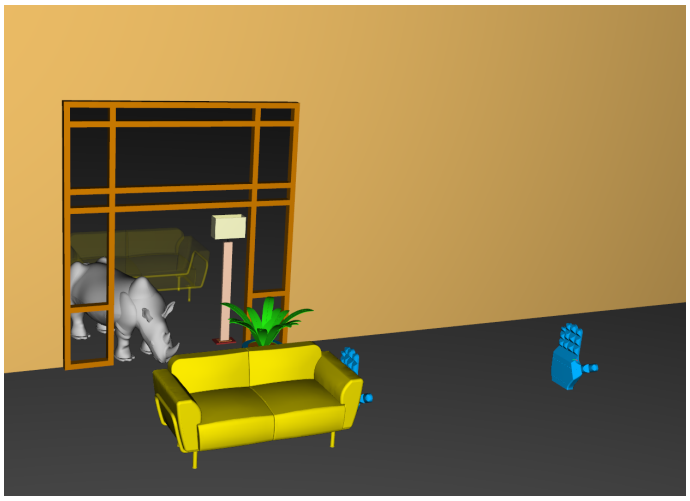
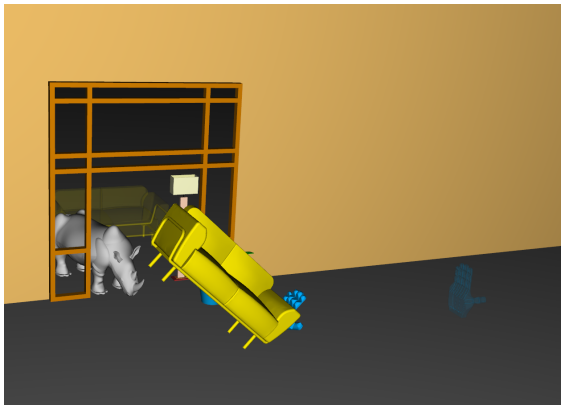
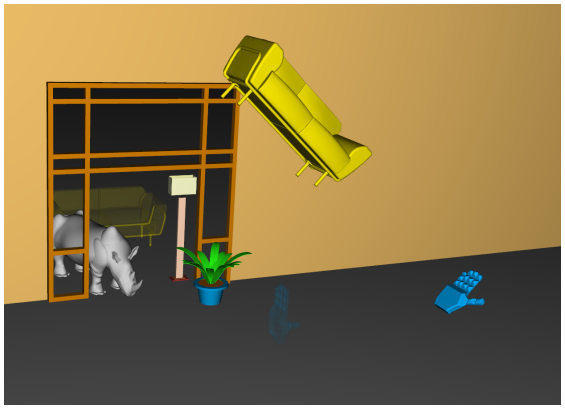


Figure 1.3: Two hands demonstrate alternative methods of controlling the motion of the couch: one physically grabbing the side of the couch, and the other manipulating it remotely through an invisible force.

the relative poses between the hands and the couch always remain constant. Consequently, inducing the same motion of the couch would require different actions from the hands. Conversely, the same 45° rotation of the hands results in vastly different motions of the couch, as depicted in Figure 1.4. This effect only applies under rotation, as translational movements of the couch are described by the same actions of the hands.



(a) The closest hand, controlling the couch.



(b) The distant hand, controlling the couch.

Figure 1.4: The couch assumes different positions when controlled by each hand, despite the fact that both hands were rotated by the same amount.

Now, suppose that the path itself should be planned in terms of the motion of one

of the hands. In that case, the choice of hand is vital, as the path is more easily described by the one closer to the couch. In general, a more easily described path should translate to a more computationally efficient problem to solve.

In the example, the hands represent anchor frames, which are coordinate systems fixed to the rigid body. In this report, it is easier to think of anchor frames as a way of controlling the rigid body from a specific position rather than as coordinate systems. Importantly, the IPS path planning algorithm describes the motion of a rigid body in terms of its anchor frame. As will be presented in the following sections, both the placement of the anchor frame and its orientation relative to the rigid body can significantly affect the performance of the path planning algorithm.

1.2 The Role of Anchor Frame Placement

Following, is an example of a seemingly trivial path planning problem where the effects of the placement of the anchor frame are apparent. Figure 1.5 illustrates a scene involving a rigid body (in orange), static geometry (in blue), and two anchor frame alternatives. The task is for the rigid body to rotate around the encapsulated

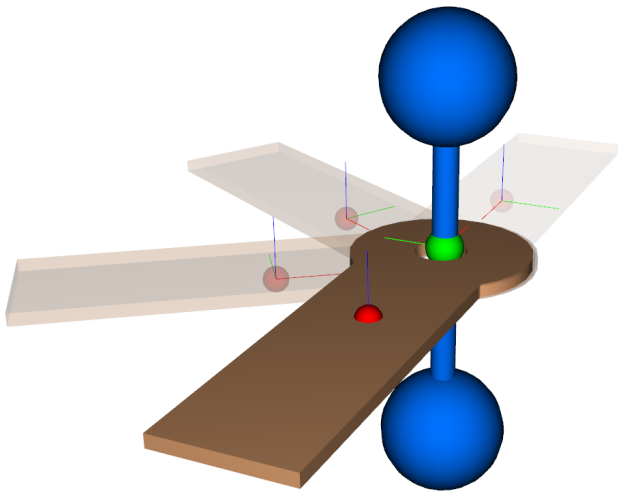


Figure 1.5: A path planning problem requiring the rigid body to rotate around the static geometry to reach its goal. The red dot represents the default anchor frame position, while the green dot in the middle of the rod marks the alternate anchor frame.

static geometry by 180°. For simplicity, the rigid body and static geometry are axis-aligned. As a result, only three dimensions in the six-dimensional configuration space are of interest: x, y, and rotations around the z-axis, denoted φ_z. Thus, all relevant configurations will reside in the subspace

$$\hat{\mathcal{C}} = \{(x, y, \varphi_z) \mid x \in \mathbb{R},\ y \in \mathbb{R},\ \varphi_z \in [0, 2\pi)\} \subset \mathcal{C}, \tag{1.1}$$

where $\mathcal{C}$ is the full six-dimensional configuration space, further discussed in Section 2.3. Every point, or configuration, in $\hat{\mathcal{C}}$ corresponds to a colliding or non-colliding state between the rigid body and static geometry.

The state of the configuration space is notably altered for different positions of the anchor frame, as illustrated in Figure 1.6. The colored points in the figure represent

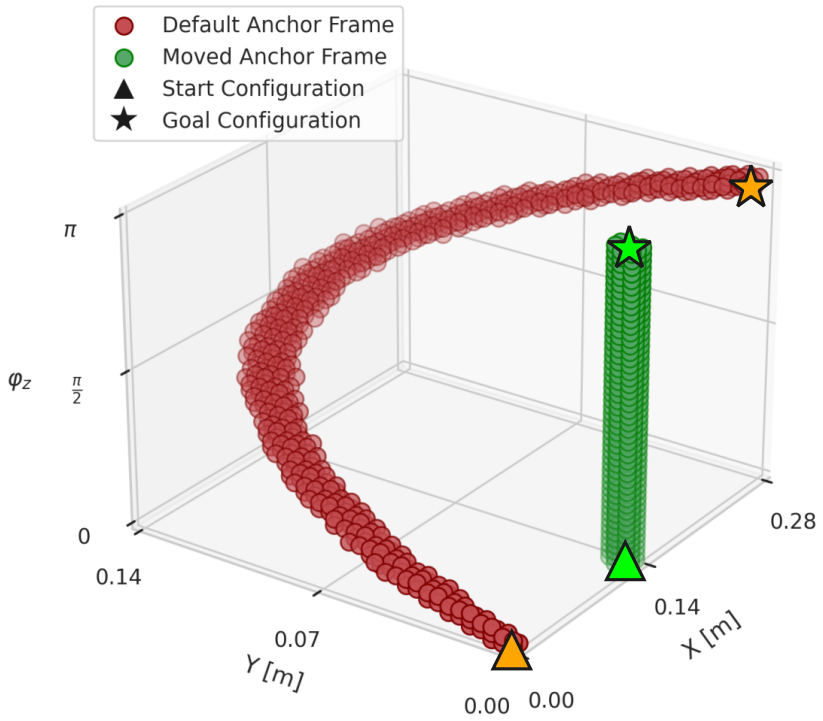


Figure 1.6: The configuration space $\hat{\mathcal{C}}$, associated with the path planning problem in Figure 1.5, for two different anchor frame placements. Points represent collision-free configurations, making up tunnels in the collision-free subspace of the configuration space. Red points correspond to the default anchor frame and green to the moved anchor frame.

non-colliding configurations, with highlighted start and goal configurations. In the physical space, solutions appear identical regardless of the anchor frame's position, and all non-colliding poses are represented in both configuration spaces. Moving the anchor frame from the default position to the middle of the rod transforms the configuration space, making it more easily navigable. This shift simplifies the path planning task, requiring only rotation around the z-axis to move from the start to the end configurations. In contrast, not moving the anchor frame requires precise

movements along all three dimensions, resulting in a much longer and more complicated path in $\hat{\mathcal{C}}$ overall. It is important to remember that the points are not evaluated beforehand, and the path must be found through sampling in all dimensions.

The state of the configuration space can have both positive and negative impacts on the performance of the path planning algorithm. FCC has not implemented any further positioning or orientation strategies for the anchor frame beyond placing it at the center of the rigid body's bounding sphere. The bounding sphere is the smallest sphere that completely encloses a model. While this strategy is favorable in most cases, FCC is interested in new strategies that enhance the path planning algorithm. Some anchor frame strategies, beyond the default placement of the anchor frame, might enable finding a path that is not currently verified to be feasible. Figure 1.7 illustrates the same rigid body and static geometry as Figure 1.5 but includes many more obstacles, making it unsolvable within typical path planning time limits when using the default anchor frame position. However, relocating the anchor frame to the middle of the rod produces a solution within seconds.

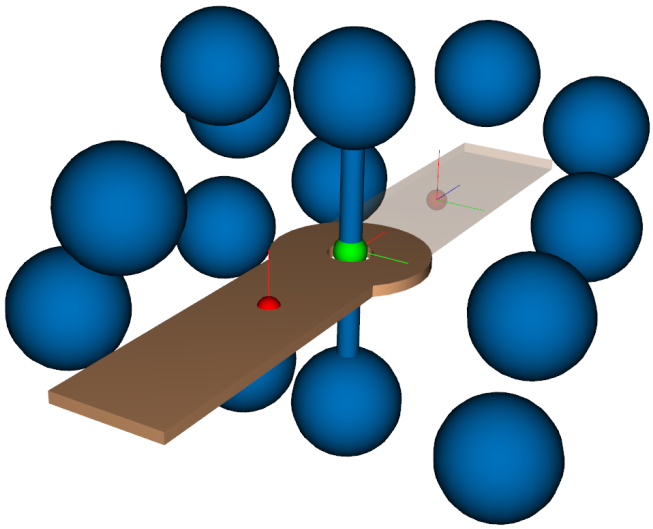
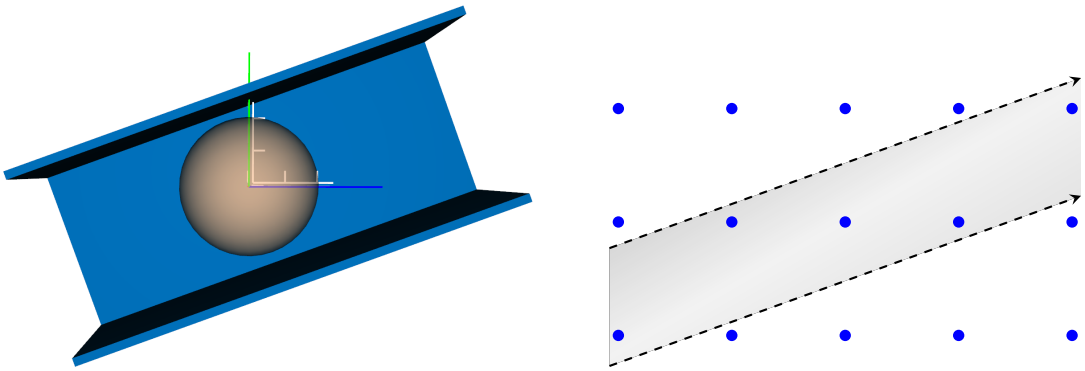


Figure 1.7: A path planning problem where the solution could not be found for the default anchor frame position, marked with a red dot. Moving the anchor frame to the green dot in the middle of the rod, yields a solution within seconds.

1.3 The Role of Anchor Frame Orientation

Unlike different placements of the anchor frame, distinct orientations do not morph or distort the configuration space. However, rotating the anchor frame also rotates

the underlying discretization, which the path planner uses to search the configuration space. Naturally, a grid more aligned with the tunnels in the configuration space, such as those depicted in Figure 1.6, provides a more easily navigated path.
Consider the spherical rigid body in Figure 1.8a, which is surrounded by a static geometry in the form of a tunnel. This scene can essentially be represented by a two-dimensional configuration subspace, depicted in Figure 1.8b.



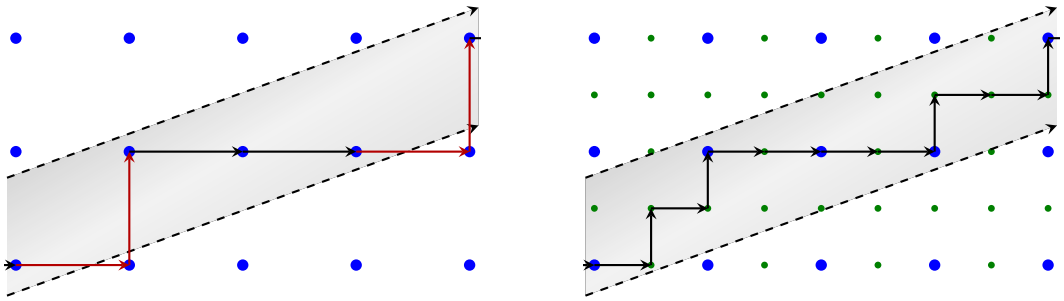
(a) A spherical rigid body in a tunnel, with which the anchor frame's axes are misaligned.

(b) Discretization over the corresponding tunnel in the configuration space.

Figure 1.8: The discretization in configuration space aligns with the axes of the anchor frame.

In a typical path planning problem, the goal would be to navigate the rigid body out of the tunnel. Given that the path planner may only take strictly horizontal or strictly vertical steps between adjacent points in configuration space [1], the current discretization allows for no feasible path through the tunnel, as illustrated in Figure 1.9a. Once the path planning algorithm has concluded that the discretization allows for no feasible path, it increases the resolution, which Figure 1.9b depicts. Since the path planner's search tree expands significantly with the increased resolution, the runtime of the path planning problem increases substantially. Suppose instead that the anchor frame is aligned with the tunnel, as in Figure 1.10a. This causes the discretization of the configuration space to align with the tunnel, as Figure 1.10b shows, implying that the path planning algorithm can find a more straightforward path without the need for higher resolution. This example highlights how a sensible anchor frame orientation can help the path planner find a solution faster.

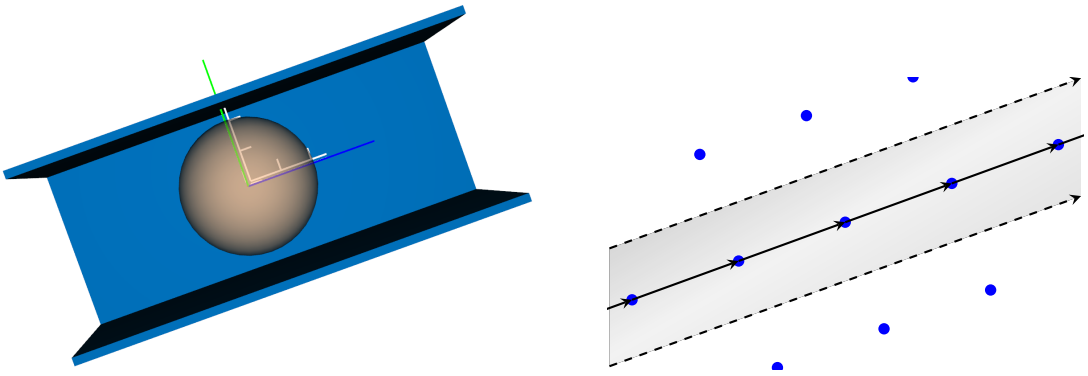
[1]The IPS Rigid Body Path Planner is actually allowed to take diagonal steps, but the points made in this section still hold.



(a) An infeasible path through the tunnel.

(b) With increased resolution, a feasible path can be found.

Figure 1.9: The path planner solves the problem of not finding a feasible path by increasing the resolution of the discretization.



(a) One of the anchor frame's axes align with the tunnel.

(b) A discretization that aligns with the direction of the tunnel allows for a simpler path, in a coarse discretization.

Figure 1.10: By aligning one of the anchor frame's axes with the physical tunnel, the discretization is aligned with the tunnel in the configuration space, resulting in a simpler path planning problem.

1.4 Scenes and Geometries

A set of geometries, loaded into the IPS software suite in the form of 3D meshes, are referred to as scenes. In this section, the specific set of scenes provided for the simulations is presented, forming the basis for testing the performance and viability of different path planning strategies. Typically, a scene consists of one or several static geometries and a rigid body. The static geometry is what serves as the obstacles for the rigid body in the path planning problem. Ideally, the developed strategies should not negatively affect path planning in any scene. However, different scenes can pose different challenges for the path planning algorithm. Some scenes might require small but purely translational movements, while others might require a mix of translational and rotational movements.

One particularly important scene is one that features a brake booster, which is to be dislodged from its surroundings. This scene has been creatively named *Brake Booster 1*. The brake booster, shown in Figure 1.11, is mounted using typical mounting studs. These studs are long rods inserted into tight mounting holes, requiring careful removal through pure translational movement. A similar scene, which also features a brake booster, has been named *Brake Booster 2* with equal creativity. This scene may not be shown due to confidentiality, but possesses the same characteristics as the *Brake Booster 1*.

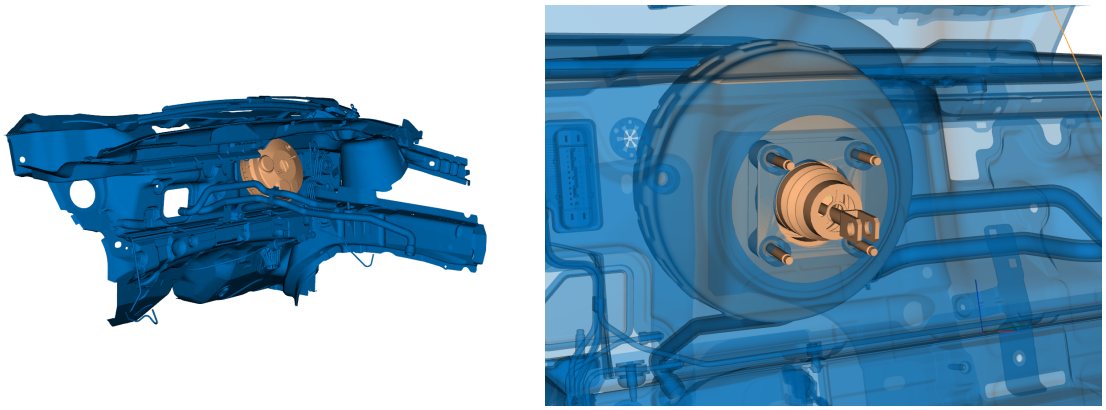


Figure 1.11: *Brake Booster 1* scene, seen in its entirety and a close-up of the four mounting studs tightly lodged into the static geometry. Model by courtesy of Volvo Cars.

For the two brake booster scenes, it has been observed that a large fraction of the total number of smallest distance computations arise from the purely translational movements involved in backing out the brake booster. Therefore, these cases are assumed to benefit the most from aligning the anchor frame with the direction of the required translation. Following is a list of all scenes used for testing in this report, along with a short description:

- *Brake Booster 1* - Installed brake booster with four mounting rods. Mainly translational movements. Depicted in Figure 1.11.

- *Brake Booster 2* - Similar to *Brake Booster 1*. Not shown due to confidentiality.
- *Center Console* - Lid for the center console in a car, mounted over the gear stick. A mix of translations and rotations. Depicted in Figure 1.12.
- *Motorcycle Carburetor* - A carburetor within the frame of a motorcycle. A mix of translations and rotations. Not shown due to confidentiality.
- *Seatbelt Retractor* - A seatbelt retractor inside the door frame of a car. A mix of translations and rotations. Later shown in Figure 4.4.

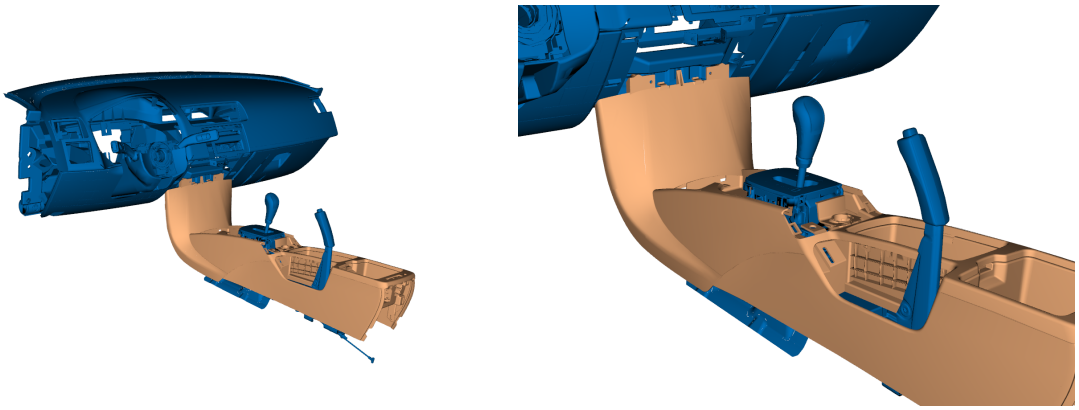


Figure 1.12: *Center Console* scene, seen in its entirety and a close-up of the challenging obstacles in the form of the gear stick, the hand brake and connections to the dashboard.

1.5 Aim

The aim of the project is to enhance the efficiency of the path planning algorithm through placement and orientation strategies for the anchor frame. The scope includes developing robust methods to measure the performance of these strategies across various initial conditions and scenes.

1.6 Limitations

Only external factors to the pre-existing path planning algorithm, such as anchor frame placement and orientation, can be used to improve its efficiency. The algorithm itself is not presented to the authors, and hence it is regarded as a black box solver. Similarly, results or methods developed during this project may not be as efficient were they implemented directly into the IPS software suite. There will be computational overhead associated with communication between the IPS software suite and its API. Performance is only measured by the number of smallest distance

computations, and not the runtime of the algorithm. This is mainly due to non-optimized code, server communication overhead and hardware dependency.

Evaluation-based optimization methods are not applicable in this project. The aim is to develop a general strategy that improves performance across various settings, rather than finding an optimal anchor frame position for specific scenarios. This approach optimizes the anchor frame position before running the algorithm, without relying on cost function evaluations. Traditional optimization typically involves minimizing a cost function (here, the number of smallest distance computations) by iteratively adjusting the decision variable (the anchor frame pose). However, this approach would only be optimal for specific static geometry, rigid body characteristics, and start/goal poses. Changes to these parameters would require re-optimization. Users of the path planning algorithm are not interested in optimal anchor frame positions for specific cases, as these provide no utility for subsequent planning scenarios. Instead, users seek to solve each path planning problem as quickly as possible before moving to the next.

1.7 Specification of the issue being investigated

It has been observed by FCC that the position and orientation of the anchor frame affects the performance of the path planning algorithm, the extent of which is to be investigated in this study. As a result, it is hypothesized that the path planning algorithm can be made more efficient by creating a more easily navigated configuration space between the start and goal configurations. This approach differs from recent research on path planning efficiency, which typically focuses on search algorithms within a given configuration space [7, 8]. The proposed methodology relies on the following two hypotheses.

Hypothesis 1: ***If*** *the motion of the path can be predicted to involve rotation around an axis,* ***then*** *positioning the anchor frame to allow a higher rotational freedom around this axis will facilitate the navigation of the configuration space*

Hypothesis 2: ***If*** *the motion of the path can be predicted to be a pure and mainly linear translation,* ***then*** *aligning one of the anchor frame's axes with the translation leads to a more easily navigated configuration space.*

A detailed description of how these hypotheses will be validated is given in Chapter 3, Methods.

2
Theory

This chapter explains the mathematical foundations of the project and establishes notation. Minor mathematical details are occasionally omitted to maintain simplicity and avoid unnecessary complexity. For a deeper understanding, readers are encouraged to refer to Chapter 2 of Murray et al.'s *A Mathematical Introduction to Robotic Manipulation* [14]. Quaternions are commonly used to represent rotations because of their numerical stability and computational efficiency [15]. This report, however, uses rotation matrices as they better align with the project's methodologies.

2.1 Rigid Body Transformations

Mathematically, a rigid body can be seen as a collection of points in $\mathbb{R}^3$ that must obey the rules of rigid body transformations.

2.1.1 Defining Rigid Body Transformations

We define a rigid body transformation similarly to Murray et al. [14, Chapter 2.1].

Definition 1 (Rigid Body Transformation)**.** A map $g : \mathbb{R}^3 \to \mathbb{R}^3$ is a rigid body transformation if the following properties are satisfied:

(i) Distances between points are preserved: $||g(\boldsymbol{p}) - g(\boldsymbol{q})|| = ||\boldsymbol{p} - \boldsymbol{q}||$ for all points $\boldsymbol{p}, \boldsymbol{q} \in \mathbb{R}^3$.

(ii) Cross products are preserved: $g(\boldsymbol{b}_1 \times \boldsymbol{b}_2) = g(\boldsymbol{b}_1) \times g(\boldsymbol{b}_2)$ for all vectors $\boldsymbol{b}_1, \boldsymbol{b}_2 \in \mathbb{R}^3$.

□

Unlike the definition made in [14], our notation for the map g remains the same whether it acts on points or vectors, as for when the homogeneous representations have been introduced in section 2.2, the operations will be computationally equivalent.

To understand why property (ii) is necessary, consider an object B undergoing the mapping $g(x, y, z) = (x, y, -z)$. This mirroring in the xy-plane preserves distances,

but not cross products. As an example, consider the vectors

$$\boldsymbol{b}_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T, \ \boldsymbol{b}_2 = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T, \text{ and } \ \boldsymbol{b}_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T.$$

With these in mind, we have

$$g(\boldsymbol{b}_1) \times g(\boldsymbol{b}_2) = \boldsymbol{b}_1 \times \boldsymbol{b}_2 = \boldsymbol{b}_3,$$

which does not equal

$$g(\boldsymbol{b}_1 \times \boldsymbol{b}_2) = g(\boldsymbol{b}_3) = -\boldsymbol{b}_3.$$

Thus, mirroring is not a rigid body transformation. This example is illustrated in Figure 2.1.

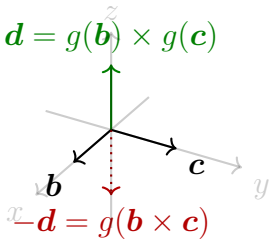


Figure 2.1: The mirroring transformation $g(x, y, z) = (x, y, -z)$ does not preserve cross products and is therefore not a rigid body transformation.

2.1.2 The Special Euclidean Group

Rigid body transformations consist of combined rotation and translation. To understand rigid body transformations fully, we first need to define the space of rotation matrices. By doing this, we can then arrive at a comprehensive definition of the space of rigid body transformations.

Definition 2 (Space of Rotation Matrices)**.** The space of rotation matrices in $\mathbb{R}^{3\times3}$ is given by

$$SO(3) = \{R \in \mathbb{R}^{3\times3} : RR^T = \mathbb{I}, \det(R) = 1\}, \tag{2.1}$$

where SO stands for *special orthogonal* and $\mathbb{I}$ is the identity matrix.

The constraint that the determinant of R must be 1, and in particular not equal to -1, ensures consistency with a right-handed coordinate frame. For further detail, see [14, Chapter 2.2.1].

With the space of rotation matrices defined, we can express the space of rigid body transformations in terms of $SO(3)$.

Definition 3 (Space of Rigid Body Transformations)**.** The set of rigid body transformations is a manifold denoted by $SE(3)$ (*Special Euclidean* group) and can be defined as

$$SE(3) := \mathbb{R}^3 \bigtimes SO(3) = \{(\boldsymbol{t}, R) : \boldsymbol{t} \in \mathbb{R}^3, R \in SO(3)\}, \tag{2.2}$$

where $\boldsymbol{t}$ represents a translation and $\bigtimes$ denotes a Cartesian product of sets.

The order in which R and $\boldsymbol{t}$ acts on a point matters, so we follow the convention that the rotation acts first, and the translation acts afterward. The rotation-translation order will be apparent in the next section, where homogeneous rigid body transformations are defined algebraically.

2.2 Homogeneous Representations

We introduce homogeneous representation of points, vectors, and transformations for its convenience and clarity. Importantly, homogeneous coordinates distinguish vectors from points algebraically, as a vector is the difference between two points in $\mathbb{R}^3$.

Points $\boldsymbol{q} \in \mathbb{R}^3$ are represented in homogeneous coordinates by appending an element of 1,

$$\bar{\boldsymbol{q}} = \begin{bmatrix} \boldsymbol{q} \\ 1 \end{bmatrix} = \begin{bmatrix} q_x \\ q_y \\ q_z \\ 1 \end{bmatrix}. \tag{2.3}$$

For vectors, the last element cancels out when subtracting one point from another, leading to the homogeneous coordinates

$$\bar{\boldsymbol{c}} = \begin{bmatrix} \boldsymbol{c} \\ 0 \end{bmatrix} = \begin{bmatrix} c_x \\ c_y \\ c_z \\ 0 \end{bmatrix}. \tag{2.4}$$

Rigid body transformations also have homogeneous representations. We denote their matrices by T such that for a point $\boldsymbol{q} \in \mathbb{R}^3$ and a rigid body transformation $g \in SE(3)$, we get

$$\begin{bmatrix} g(\boldsymbol{q}) \\ 1 \end{bmatrix} = T\bar{\boldsymbol{q}} = T \begin{bmatrix} \boldsymbol{q} \\ 1 \end{bmatrix}. \tag{2.5}$$

The transformation is partitioned as

$$T = \begin{bmatrix} R & \boldsymbol{t} \\ 0 & 1 \end{bmatrix}, \tag{2.6}$$

where $\boldsymbol{t} \in \mathbb{R}^3$ represents a translation and $R \in SO(3)$ represents a rotation about the origin of the coordinate system. One can verify that a transformation matrix T as in (2.6) satisfies (2.5), by observing that $R\boldsymbol{q} + \boldsymbol{t}$ is indeed a rigid body transformation of $\boldsymbol{q}$. Therefore, the 4-by-4 matrix T is the homogeneous representation of $g \in SE(3)$.

A significant advantage of rigid body transformations is their concise description of a pose. Given a well-defined rigid body with a local coordinate frame, each state of the rigid body can be described in matrix form (2.6). The vector $\boldsymbol{t}$ represents the placement of the local coordinate frame (and thus the rigid body), while the matrix R represents its orientation. In other words, $SE(3)$ can also be seen as the configuration space of the rigid body. Figure 2.2 shows a rigid body transformation, described by matrix multiplication of homogeneous rigid body transformations.

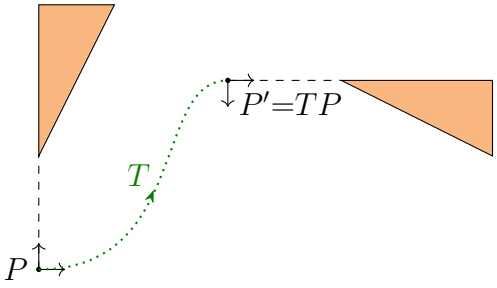


Figure 2.2: A rigid body transformation defined by matrix multiplication. The dotted path is, however, not defined by the transformation T, as T only maps the frame from pose P to P'.

2.3 Configurations, Poses, and Transformations

This section aims to distinguish closely related terms with subtle differences. Understanding these distinctions is helpful for grasping the nuances of rigid body path planning. The configuration space is a fundamental concept in path planning [16]. In this report, it refers to the space of all possible positions and orientations of the rigid body, where each point corresponds to a unique pose of the rigid body. The configuration space can be parameterized in many ways, so the description given here is not exclusive.

The configuration space extends to six dimensions to account for both the position and orientation of the rigid body. This allows for a complete mapping of all possible states the rigid body can assume, creating a mathematical space where path planning becomes more tangible. In path planning, it is common practice to parameterize

the position of the rigid body with Cartesian coordinates $(x, y, z) \in \mathbb{R}^3$, while the parameterization of the orientation varies more in literature. One way of extending the configuration space to six dimensions, similar to the two-dimensional example in Equation (1.1), is by choosing

$$\mathcal{C} = \mathbb{R}^3 \bigtimes \mathbb{S}^3, \tag{2.7}$$

where

$$\mathbb{S}^3 = \{\boldsymbol{x} \in \mathbb{R}^4 : ||\boldsymbol{x}|| = 1\}. \tag{2.8}$$

The set $\mathbb{S}^3$ is known as the three-sphere, and is often used to describe quaternion based motion modeling [17, 18]. $\mathbb{S}^3$ double-covers $SO(3)$ and, as LaValle thoroughly explains in [18, Chapter 4], by identifying antipodal points, $\mathbb{S}^3$ could be declared topologically equivalent to $SO(3)$ due to their *homeomorphism*. An alternative description of configuration space is thus obtained by replacing $\mathbb{S}^3$ by $SO(3)$ in Equation (2.7), yielding

$$\mathcal{C} = \mathbb{R}^3 \bigtimes SO(3) = SE(3), \tag{2.9}$$

where the second equality defines $SE(3)$ in (2.2). Alternative descriptions exist, such as the one given by LaValle [18, Chapter 4][1], but (2.7) and (2.9) are two definitions that work well with the nomenclature of this report.

Now that the configuration space is defined, a *configuration* refers to an element in that space. While a configuration denotes a point in an abstract configuration space, a *pose* refers to the position and orientation of the rigid body in physical space. In particular, the pose of a rigid body is merely an intuitive way to describe the state of a rigid body as one would see it in real life. A *transformation*, on the other hand, is a function which maps a pose to a new one. It can be considered mathematically, e.g., as an element of $SE(3)$, or qualitatively as a combined translation and rotation of an object.

To summarize, a configuration $\zeta \in \mathcal{C}$ can be considered as a point in the abstract configuration space $\mathcal{C}$. Both poses P and transformations T belong to the same space, $P, T \in SE(3)$, but a pose should be understood as a specific position and orientation of the rigid body, whereas a transformation maps poses and configurations to new states. Specifically, it acts on configurations if the configuration is first expressed in its corresponding pose. However, these distinctions are not crucial for the methods; they are only meant to clarify the meaning of each term.

2.4 Rigid Body Motion

To describe rigid body motions, it can be helpful to first consider a description of the motion of a three-dimensional object without rotations. Suppose that the

[1] LaValle describes $\mathcal{C} = \mathbb{R}^3 \bigtimes \mathbb{RP}^3$, but acknowledges the feasibility of using $\mathcal{C} = SE(3)$ by noting their homeomorphism. The manifold $\mathbb{RP}^3$ is called the *real projective space* [19] and can be thought of as a unit sphere $\mathbb{S}^3 \subset R^4$ with antipodal (opposite) points considered equivalent.

rigid body, originally in position $\boldsymbol{p} \in \mathbb{R}^3$, is to travel to a new position $\boldsymbol{q} \in \mathbb{R}^3$. The motion, which in this case is a pure translation, is defined by $m+1$ positions $\boldsymbol{u}_i, \quad i = 0, \dots m$ such that $\boldsymbol{u}_0 = \boldsymbol{p}$ and $\boldsymbol{u}_m = \boldsymbol{q}$, with m interpolations between these positions. Thus, the motion of of the rigid body can be described as

$$\boldsymbol{u} = \boldsymbol{u}_i + \eta(\boldsymbol{u}_{i+1} - \boldsymbol{u}_i), \quad \eta \in [0,1], \quad i \in \{0, \dots, m-1\}.$$

A vector $\boldsymbol{u}_{i+1} - \boldsymbol{u}_i$ can be seen as the resulting mapping from $\boldsymbol{u}_{i+1} - \boldsymbol{u}_i$, and by scaling this mapping from zero to one, a motion is simulated. Unfortunately, describing motion where rotation involved is not as straightforward. This will be further discussed in the coming sections.

2.4.1 Interpolation

When the rigid body path planning algorithm searches the configuration space, it measures clearance at every point. Clearance is the smallest distance between the rigid body and the static geometry. Often, the sampled configuration will be in collision, resulting in a clearance of zero. However, positive clearance provides a safe region within the six-dimensional configuration space. If another sample with positive clearance is found nearby and their regions overlap, a linear interpolation between the two configurations is assumed to be possible. The exact implementation in the IPS software suite is unknown to the authors, and mathematical details about rigid body interpolation are not provided. A path is established only when a chain of connected samples, with overlapping clearances, is formed between the start and goal configurations.

When the IPS path planning algorithm returns a complete or partial path, it provides a discrete sequence of non-colliding connected rigid body poses, known as waypoints. Although adjacent waypoints are typically close, describing the path as continuous is necessary. The IPS software suite achieves this through shortest-distance interpolation in the configuration space, ensuring all poses along the interpolation are collision-free.

Transformation interpolation techniques are a significant scientific field [20, 21], but they are not central to this project. Instead, the key takeaway is that waypoint-to-waypoint interpolation can be considered to have constant translational and rotational velocities [22, Section 1]. This understanding sets the stage for exploring the control of these velocities in greater detail.

2.4.2 Twists and Screws

This section mathematically connects rigid body transformations and motions with instantaneous motion properties, specifically velocities. Although velocities are generally time-dependent variables and are treated as such in most motion planning literature, they will be regarded as constant throughout this section. Omitting the time dependencies emphasizes our focus on isolated points in time, highlighting the

quasi-static nature of the rigid body transformations considered in this project.

Consider Figure 2.3, depicting a point $\boldsymbol{p} \in \mathbb{R}^3$ on a rigid body in a given configuration. The instantaneous movement, which consists of a translational velocity $\boldsymbol{v}$ and an angular velocity $\boldsymbol{\omega}$, is defined with respect to the anchor frame position $\boldsymbol{o}_0$. Thus, the rotational axis goes through $\boldsymbol{o}_0$ and is parallel to the z-axis, i.e., perpendicular to the image plane. The alignment with the z-axis is chosen for simplicity but causes no loss of generality. The differential equation describing the motion of $\boldsymbol{p}$ is then

$$\dot{\boldsymbol{p}} = \boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{b}_0, \tag{2.10}$$

with $\boldsymbol{b}_0 = \boldsymbol{p} - \boldsymbol{o}_0$.

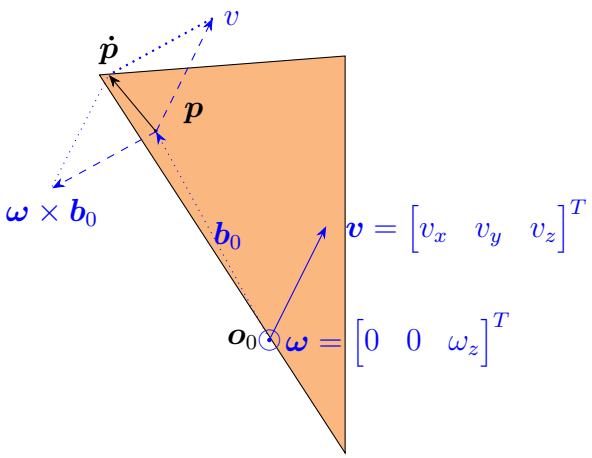


Figure 2.3: The point $\boldsymbol{o}_0$ is the center of rotation. The dashed blue lines are the instantaneous velocity components acting on $\boldsymbol{p}$, resulting in the solid, black vector that is $\dot{\boldsymbol{p}}$.

Now that the velocity components are introduced, it is time to define an important concept in kinematics, known as a twist.

Definition 4 (Twist)**.** Given a translational velocity $\boldsymbol{v} \in \mathbb{R}^3$ and an angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ around a point $\boldsymbol{o}_0 :\in \mathbb{R}^3$, the corresponding *twist* is defined as

$$\bar{\boldsymbol{\xi}} = \begin{bmatrix} \hat{\omega} & \boldsymbol{v} - \boldsymbol{\omega} \times \boldsymbol{o}_0 \\ \boldsymbol{0} & 0 \end{bmatrix}, \tag{2.11}$$

where $\hat{\omega}$ is the "cross product matrix" parameterized by $\boldsymbol{\omega}$,

$$\hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \implies \hat{\omega}\boldsymbol{q} = \boldsymbol{\omega} \times \boldsymbol{q}, \ \forall \boldsymbol{q} \in \mathbb{R}^3. \tag{2.12}$$

□

The twist is parameterized by

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} - \boldsymbol{\omega} \times \boldsymbol{o}_0 \end{bmatrix}. \tag{2.13}$$

In the rest of the report, a *twist* refers to both the twist in parameterized form, also known as exponential coordinates, and the twist in matrix form. The bar notation ($\bar{\xi}$) is specifically used to indicate the matrix form. Twists mathematically relate to rigid body transformation in the following way. For any twist $\bar{\xi}$ defined as above and for any scale parameter [2] $\theta \in \mathbb{R}_{\geq 0}$, the expression

$$T = \exp\left(\bar{\xi}\theta\right) \tag{2.14}$$

is an element of $SE(3)$, and thus represents a rigid body transformation.

Just like rigid body transformations, twists are defined with respect to a rotational axis. In general, the translational and rotational components of a twist can have different directions, leading to the concept of screw coordinates. Mozzi-Chasles' theorem [23, 24] [3] says that any rigid body transformation $\exp(\bar{\xi}\theta)$ can be expressed as a screw motion, which involves rotating about and translating along a common axis. In other words, a suitable choice of rotational axis means that the rigid body spirals around the axis on the path $\exp(\bar{\xi}\tau)$, $\tau \in [0, \theta)$. Defining the point $\boldsymbol{o}_1$ as

$$\boldsymbol{o}_1 = \boldsymbol{o}_0 + \frac{\boldsymbol{\omega} \times \boldsymbol{v}}{||\boldsymbol{\omega}||^2}, \tag{2.15}$$

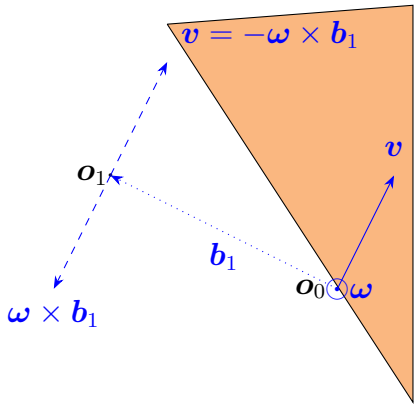
the new screw axis is given by

$$\boldsymbol{o}_1 + \boldsymbol{\omega}\lambda, \quad \lambda \in \mathbb{R}. \tag{2.16}$$

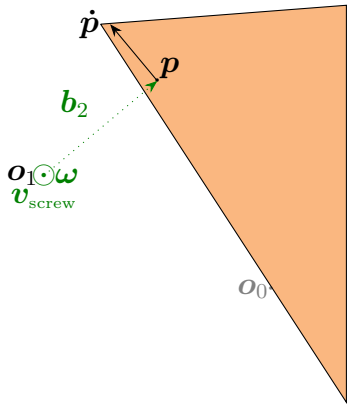
Figure 2.4a shows that at the point $\boldsymbol{o}_1$, the rotational movement is canceled out by v_x and v_y. With the rotational axis given by the formula (2.16), the translational velocity $\boldsymbol{v}$ is replaced by its projection onto $\boldsymbol{\omega}$, that is, $\boldsymbol{v}_{\text{screw}} = \boldsymbol{\omega}\left(\frac{\boldsymbol{v}\cdot\boldsymbol{\omega}}{||\boldsymbol{\omega}||^2}\right)$, which in the case of $\boldsymbol{\omega} = \begin{bmatrix} 0 & 0 & \omega_z \end{bmatrix}^T$ means that $\boldsymbol{v}_{\text{screw}} = \begin{bmatrix} 0 & 0 & v_z \end{bmatrix}^T$, as depicted in Figure 2.4b.

[2] One could actually extend this expression to $\theta \in \mathbb{R}$, but since θ often is thought of as a time variable, it is suitable to treat it as a non-negative parameter.

[3] The theorem is often attributed to either Mozzi (as in [25]) or Chasles (as in [14, Chapter 2]), who discovered the result independently.



(a) The dashed, blue arrows represent the velocity components acting on the point $\boldsymbol{o}_1$, had it been part of the rigid body. The two velocity components cancel each other out in the plane of the image, leaving only $\begin{bmatrix} 0 & 0 & v_z \end{bmatrix}$ active.



(b) The rotational axis has been moved so that it goes through $\boldsymbol{o}_1$. The reader can verify that $\dot{\boldsymbol{p}}$, now expressed as $\dot{\boldsymbol{p}} = \boldsymbol{v}_{\text{screw}} + \boldsymbol{\omega} \times \boldsymbol{b}_2$, is the same as in Figure 2.4a. Since $\boldsymbol{p}$ is an arbitrary point on the rigid body, it means that the velocities of all other points on the rigid body are also the same.

Figure 2.4: By finding a point that instantaneously has no velocity perpendicular to the angular velocity, one finds the screw axis. In this example, the screw axis does not intersect with the rigid body.

3
Methods

Basic setups and procedures are initially described in the chapter. These include explanations of data acquisition methods, the concept of evenly distributed orientations, and the fundamental algorithm for stepwise path planning. The chapter then outlines the two main phases which the project was conducted in. The first phase was exploratory, focusing on quantifying the impact of the anchor frame's position and orientation on overall performance. Insights from this phase guided the development of strategic methods to enhance the efficiency of the path planning algorithm. The impact evaluation and strategies are discussed in separate sections of the chapter.

3.1 Basic Setup and Procedures

A few techniques and ideas have been used throughout the study. Together, they form a basic framework on which both the anchor frame pose impact assessment and the implemented strategies have been built.

3.1.1 Stepwise Path Planning

Stepwise path planning was used to dynamically adjust the anchor frame pose during the path planning algorithm's execution. In early experiments not disclosed in this report, it was found that setting the anchor frame once for the entire path proved difficult. It makes more sense to update the anchor frame pose to be suitable for imminent movements instead. The API allows enabling a setting that forces the path planner to abort once certain progress conditions are met. While the exact details of these conditions are unknown to the authors, the general principle is that the path planner stops when greater clearance is found along a path. Upon stopping, the algorithm returns a sub-path composed of waypoints.

From the perspective of the path planning algorithm, each time the path was aborted and recommenced, a new path planning problem began. This provided an opportunity to alter the pose of the anchor frame between each step. To continue planning the next step, the rigid body was moved to the last waypoint of the previous step, which then served as the new starting position for the next step. Algorithm 1 describes stepwise path planning in pseudocode.

Algorithm: Step-wise Path Planning
Purpose:
- Compute a path for a rigid body from start to end configuration while avoiding collisions with static geometry.

Inputs:
- Rigid body with start and end configurations
- Static geometry

Outputs:
- Path from start to end configuration

Procedure:
1. Initialize the path planner
   (a) Set start configuration
   (b) Set goal configuration
   (c) Set `pathFound` to `false`
   (d) Initialize list for sub-paths
2. **WHILE** `pathFound` is `false`
   (a) Adjust anchor frame placement and/or orientation
   (b) Call path planner to plan a step towards the goal
   (c) Move rigid body to the last waypoint of the planned step
   (d) **IF** the rigid body has reached the goal configuration **THEN**
   Set `pathFound` to `true`
   ELSE
   Set new start configuration from current pose
   (e) Append sub-path to output list
3. Concatenate all sub-paths and return final path

Algorithm 1: Step-wise path planning. The strategies take place in step 2(a).

3.1.2 Data Acquisition

Performance measurements, essential for both the impact assessment phase and strategy evaluation, are based on the number of times the path planning algorithm calculates the smallest distance between the rigid body and the static geometry. During the algorithm's runtime, rigid body configurations are sampled through temporary transformations. From each sampled pose, the smallest distance between the rigid body and the static geometry is calculated, as illustrated in Figure 3.1. In every sampled pose, a pair of points is produced: one on the rigid body, denoted as $\tilde{\boldsymbol{p}}$, described relative to its local object frame, A_B, and the other on the static geometry, denoted as $\boldsymbol{q}$, specified in global coordinates. These points represent where, on the two bodies, the shortest distance was found. Note that each step does not necessarily produce a unique pair of points, as one pair can correspond to the shortest distance for several poses in the path. Given a point pair $\tilde{\boldsymbol{p}}$ and $\boldsymbol{q}$, we define their collision vector $\boldsymbol{c}$ as $\boldsymbol{c} = \boldsymbol{q} - \boldsymbol{p}$, where $\boldsymbol{p}$ is $\tilde{\boldsymbol{p}}$ in global coordinates. The name "collision vector" is suitable since $\boldsymbol{c}$ is the shortest translation of the rigid body that leads to collision with the static geometry.

As specified in the description of stepwise path planning in Section 3.1.1, after

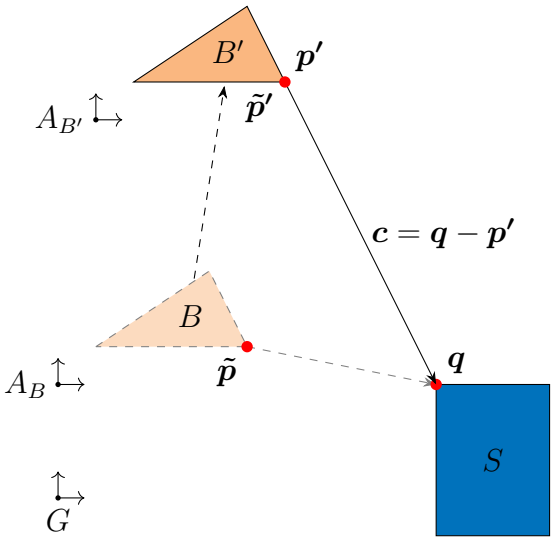


Figure 3.1: Diagram of a rigid body, B, adjacent to static geometry, S. In its original position, the smallest distance computation identifies the local point $\tilde{\boldsymbol{p}}$ and the global point $\boldsymbol{q}$. The point $\tilde{\boldsymbol{p}}$ is defined relative to the rigid body's anchor frame A_B. B' shows the rigid body at the final waypoint of the planning step, with $\boldsymbol{p'}$ being $\tilde{\boldsymbol{p}}'$ expressed in global coordinates. The collision vector $\boldsymbol{c}$ is depicted between $\boldsymbol{p'}$ and $\boldsymbol{q}$.

a step has been successfully planned, the rigid body is transformed to the starting configuration of the subsequent step, labeled B'. The decision was made to calculate the collision vectors, from this new position, rather than extracting them from the position at the time of measurement (i.e., using point $\boldsymbol{p'}$ instead of $\boldsymbol{p}$ in Figure 3.1). Since the measurements on the rigid body remain consistent under transformations, moving the rigid body in the direction of the collision vector will still lead to a collision. However, the collision may occur before the rigid body has moved the entire distance of the collision vector. Despite this, the critical information about the potential collision direction remains accurate and useful.

As previously explained, rigid body poses are manipulated through their anchor frames, which remain fixed relative to the rigid body itself. Consequently, expressing $\tilde{\boldsymbol{p}}'$ relative to $A_{B'}$ is equivalent to expressing $\tilde{\boldsymbol{p}}$ relative to A_B. Therefore, the global point $\boldsymbol{p'}$ can be articulated as $\boldsymbol{p'} = A_{B'}\tilde{\boldsymbol{p}}' = A_{B'}\tilde{\boldsymbol{p}}$. Thus, the collision vector $\boldsymbol{c}$ is computed as

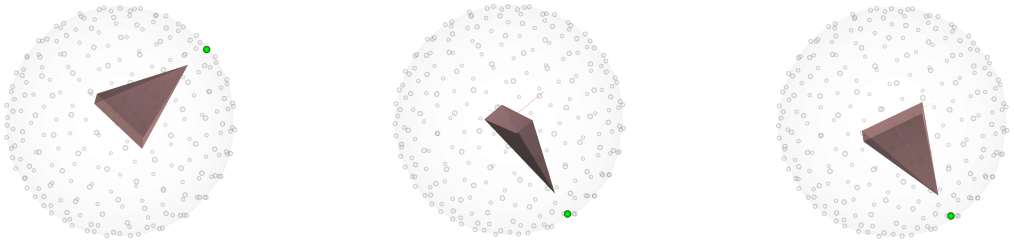
$$\begin{aligned} \boldsymbol{c} &= \boldsymbol{q} - \boldsymbol{p'} \\ &= \boldsymbol{q} - A_{B'}\tilde{\boldsymbol{p}}', \end{aligned}$$

where $A_{B'}$, $\tilde{\boldsymbol{p}}$ and $\boldsymbol{q}$ all can be directly extracted from the path planner at the end of a step. The number of available $\boldsymbol{q}, \tilde{\boldsymbol{p}}$-samples from a step is equal to the number of smallest distance computations for that step.

3.1.3 Evaluation and Evenly Distributed Orientations

It is already known that the initial orientation of the anchor frame relative to the geometry affects the algorithm's performance. This report investigates the extent of that effect. Since users typically do not account for the initial orientation, and should not need to for usability, any initial orientation can be expected. Therefore, a successful strategy should enhance the general performance across a wide range of initial orientations.

To approximate a spectrum of arbitrary orientations with an even distribution, the following systematic approach was adopted. Firstly, 256 evenly distributed points were generated on the unit sphere using spherical Fibonacci mapping [26]. The anchor frame was then rotated so that one of its axes, typically the z-axis, pointed towards each of the generated points. For each point, the anchor frame was further rotated around the z-axis by 120 degrees two times, completing a full rotation. The method generates $256 \times 3 = 768$ rotation matrices, $R_i \in SO(3),\ i \in [1, \ldots, 768]$, which operate on the initial default anchor frame pose. This process is illustrated in Figures 3.2a through 3.2c, with the pyramid illustrating the anchor frame in this example, as opposed to the rigid body. The idea is that the rigid body is fixed such that the anchor frame rotates relative to it.



(a) The anchor frame's z-axis, here highlighted by the sharp point of a pyramid, points at one of 256 evenly distributed points on the unit sphere.

(b) The anchor frame's z-axis points towards each of the generated points.

(c) For each point, the anchor frame's $x-$ and y-axes are rotated around the z-axis by 120° twice, completing a full rotation.

Figure 3.2: The process of generating 768 unique and evenly distributed orientations. 256 points are generated on the unit sphere, an axis of the rigid body's anchor frame is pointed towards each point, then additionally rotated twice by 120°.

The strategies developed were benchmarked by running the same path planning problem using each of the 768 initial orientations. This approach effectively eliminates the uncertainty of whether a strategy is superior only for a selected number of initial orientations.

3.2 Impact Assessment of Anchor Frame Pose

It is important to determine the extent to which different poses of the anchor frame impact the performance of the path planning algorithm. This section details the methods used to collect performance data based on varying anchor frame positions and orientations.

3.2.1 Orientation Impact

As discussed previously in Section 1.2, the orientation of the anchor frame is crucial for aligning the discretization with collision-free "tunnels" in the configuration space. While the discretization is relative to the anchor frame, the translations required for the rigid body to reach the goal position can be expressed globally. Therefore, the alignment of the discretization with these translations, and consequently the tunnels in the configuration space, depends on the initial orientation of the anchor frame. This implies that the performance of the path planning algorithm is also influenced by this initial orientation.

To measure a wide range of orientations, the method described in Section 3.1.3 was used for the initial orientations of the anchor frame. The orientations generated by this method are given by rotation matrices, $R_i \in SO(3),\ i \in \{1, \ldots, 768\}$. The pose of the anchor frame for each orientation, denoted P_i, was set to

$$P_i = \begin{bmatrix} R_i & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix}.$$

After setting the pose, stepwise path planning was run without further interference. The number of smallest distance computations was measured for each pose and recorded as the metric of performance. The evenly distributed orientations are meant to capture any initial orientation that a user of the path planning algorithm might use.

3.2.2 Placement Impact

As discussed in Section 1.2, the placement of the anchor frame alters the state of the configuration space. Some positions of the anchor frame yield a more navigable configuration space, while others might make it more difficult to traverse. The ease of navigating this configuration space should be reflected in the performance. By moving the anchor frame to various positions and measuring the performance for each one, the overall impact can be established.

To generate multiple candidate anchor frame positions within a reasonable distance from the original anchor frame, the following method was employed. The bounding box, which is the smallest axis-aligned box that encapsulates the rigid body, was partitioned into $12 \times 12 \times 12 = 1728$ cuboids of equal dimensions. The center points of the cuboids formed a cubic lattice, generating positions $\boldsymbol{t}_j \in \mathbb{R}^3,\ j \in \{1, \ldots, 1728\}$.

Similar to anchor frame orientation, the pose of the anchor frame for each position, P_j, was set to

$$P_j = \begin{bmatrix} R & \boldsymbol{t}_j \\ \mathbf{0} & 1 \end{bmatrix}.$$

For each position, after setting the anchor frame, stepwise path planning was executed, and the number of smallest distance computations was recorded as the performance metric. A significant issue with solely varying the placement is the use of only one orientation, which may not represent the full set of possible orientations. Ideally, to mitigate this variance, all evenly distributed orientations described in Section 3.1.3 would be used for each anchor frame position, with results aggregated for that position. However, this approach would lead to infeasible runtimes, as path planning on the given set of scenes take very long for poor anchor frame placements. Running all anchor frame positions for just one orientation takes roughly one full day, making it infeasible to use all 768 initial orientations.

To address this, the dependency is acknowledged, and only two complete runs of all anchor frame positions with different initial orientations are performed. This approach allows for assessment of the impact of anchor frame placement, while also providing some insight into how varying initial orientations might influence this impact.

3.3 Anchor Frame Orientation Strategy

This strategy investigates the hypothesis that aligning one axis of the rigid body's anchor frame with the predicted translation of the subsequent sub-path will yield a more navigable configuration space. The main outline of the method is simple:

1. **Gain insights on the forthcoming translation**: Use available data to predict the direction of the rigid body's movement in the next step.

2. **Align one of the anchor frame's axes with the predicted translation**: Adjust the orientation of the anchor frame to facilitate easier navigation through the configuration space.

While the method may seem straightforward, predicting future translation is challenging. An important early experiment used a preview step to gain necessary insights into future translation. By allowing the path planner to plan one step ahead and then returning to the initial position as if no planning had been done, perfect information about the future path was obtained. The "predicted" translation was simply determined as the displacement vector between the rigid body's current position and the final waypoint of the planned path in the preview step. It should be noted that this direction is in many cases not the best possible solution. A path is seldom completely straight, containing many intricate twists and turns. Furthermore, this strategy requires planning the entire path twice, and is therefore not a

viable strategy. However, promising results indicated that a good strategy exists, and the challenge was to achieve similar results by predicting the translation only from available data.

A model was developed to predict the direction of the next step, utilizing the collision vectors discussed in Section 3.1.2 and generating initial measurements when no collision vectors were available. The core idea of using available collision vectors to predict future translation was maintained throughout the project, though the details were refined through much trial and error.

3.3.1 Initial Measurements

To make predictions about the initial direction of translation when no previous path planning steps are available, complementary measurements were necessary. Points were evenly distributed on the unit sphere using the Fibonacci spherical mapping, defining directions for temporary translations of the rigid body. At the starting position, an initial clearance was measured to establish the minimum distance between the rigid body and any static geometry. This clearance ensures that translations along any sampled direction do not result in collision. For each tested direction, the rigid body was translated and measured. Consequently, each direction sampled on the unit sphere produced a new collision vector.

Similar to the collision vectors collected from the path planning algorithm, these vectors were expressed relative to the starting position of the subsequent step, which, in this initial step, is the actual starting position of the problem. The measurements made in generating the vectors were included in the total number of smallest distance computations. However, the number of directions needed for translating the rigid body did not need to be large to gather diverse collision vectors. We selected 64 evenly distributed directions of clearance length as collision vectors, and since this process occurs only once when solving a path planning problem, it is not very costly in terms of smallest distance computations. Figure 3.3 illustrates perturbing the rigid body inside the static geometry, each time generating a new collision vector.

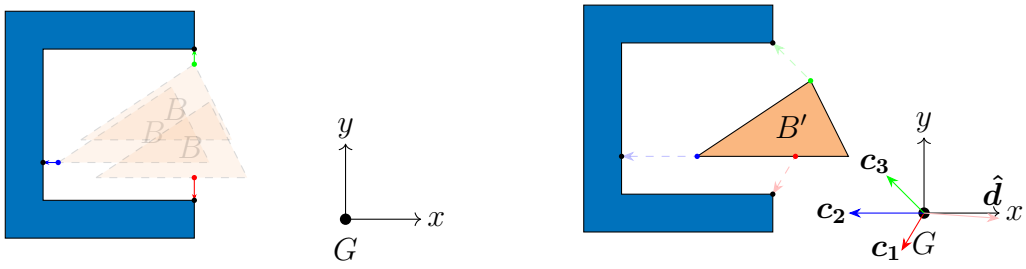
(a) Perturbations of the rigid body, yielding new smallest distance measurement point-pairs for each translation.

(b) Collision vectors, constructed from the smallest distance measurement point-pairs, drawn for the rigid body in its starting position.

Figure 3.3: Process of generating initial measurements by making small adjustments to the position of the rigid body, yielding collision vectors from the starting position.

3.3.2 Local Translation Prediction

As described in Section 3.1.2, clearance measurements are made by the path planner during the planning of a step, as illustrated in Figure 3.4a. A set of collision vectors, $\{\boldsymbol{c}_i\}_{i=1}^{N}$, is obtained at the starting position of the next step, where the rigid body in Figure 3.4b is represented by B'. Naturally, a direction of translation that minimizes alignment with the collision vectors is favorable. As long as the angle between the



(a) Measurements are sampled during the planning of a step.

(b) The predicted translation $\hat{\boldsymbol{d}}$ is determined by optimization based on the collision vectors, which are calculated based on the starting position of the next step.

Figure 3.4: Depiction of the optimization process. A predicted translation is found based on collision vectors gathered from the preceding planning step.

direction of translation and any collision vector is at least 90 degrees, moving the rigid body along the direction of translation will not result in collision, with respect to that particular collision vector. For a predicted direction of translation $\hat{\boldsymbol{d}}$, one could use $\{\boldsymbol{c}_i \cdot \hat{\boldsymbol{d}} \leq 0\}_{i=1}^{N}$ as a set of constraints. Through experience, it has been

found that such constraints typically result in an empty feasible region. Instead, the sum of the dot products between a candidate direction and each normalized collision vector was treated as a penalty function, and an optimization problem was formulated as:

$$\begin{aligned} \underset{\hat{\boldsymbol{d}}}{\text{minimize:}} \ & \sum_{i=1}^{N} \frac{\hat{\boldsymbol{d}} \cdot \boldsymbol{c}_i}{\|\boldsymbol{c}_i\|} \\ \text{subject to:} \ & \|\hat{\boldsymbol{d}}\| = 1. \end{aligned} \tag{OP1}$$

In Problem (OP1), N is the number of smallest distance computations in the previous step. The constraint that the direction $\hat{\boldsymbol{d}}$ should be of unit length eliminates the risk of the solution becoming $\mathbf{0}$. It is important to note that the optimization process is not the main focus of this study. It is merely a tool to quickly and inexpensively find a suitable direction based on the collision vectors. A few other penalty functions were tested and compared, and the one that gave the best and most consistent results was selected. The other penalty functions considered were the following:

1. Maximizing the smallest angle between $\hat{\boldsymbol{d}}$ and any collision vector $\boldsymbol{c}_i$:

$$\underset{\hat{\boldsymbol{d}}}{\text{maximize:}} \ \min_i \left(\arccos \left(\frac{\hat{\boldsymbol{d}} \cdot \boldsymbol{c_i}}{\|d\| \|\boldsymbol{c_i}\|} \right) \right) \tag{OP2}$$

2. Minimizing the count of positive dot products,

$$\underset{\hat{\boldsymbol{d}}}{\text{minimize:}} \ \sum_{i=1}^{N} I(\hat{\boldsymbol{d}}, \boldsymbol{c_i}) \tag{OP3}$$

   where $I(\hat{\boldsymbol{d}}, \boldsymbol{c_i})$ is defined as

$$I(\hat{\boldsymbol{d}}, \boldsymbol{c_i}) = \begin{cases} 1 & \text{if } \ \hat{\boldsymbol{d}} \cdot \boldsymbol{c_i} > 0 \\ 0 & \text{otherwise} \end{cases} .$$

3. Using the objective function of (OP1), but replacing $\|\boldsymbol{c}_i\|$ by $\|\boldsymbol{c}_i\|^2$ in the denominator of each term.

The core idea behind these objective functions remains the same: finding a direction that minimizes movement in the directions of the collision vectors. Initially, it might seem more intuitive to divide by the squared norm of the collision vectors, as in penalty function 3, since this approach would assign a greater weight to shorter (and thus more urgent) collision vectors. However, heuristic evidence suggests that better results are achieved by dividing only by the norm, thereby ignoring the length of the collision vectors. The unexpected improvement from ignoring the length of the collision vectors may be due to avoiding overfitting to the shortest vectors, making the direction more adaptable to various data.

The optimization problem in (OP1) has a simple analytical solution, namely,

$$\hat{\boldsymbol{d}} = -\sum_{i=1}^{N} \frac{\boldsymbol{c}_i}{||\boldsymbol{c}_i||} \Bigg/ \left|\left| \sum_{i=1}^{N} \frac{\boldsymbol{c}_i}{||\boldsymbol{c}_i||} \right|\right|, \tag{3.1}$$

but the optimization problems tested throughout the project generally require a numerical solver. For this purpose, an optimization solver was selected based on simplicity, speed, and stability. Sequential Least Squares Programming (SLSQP) was employed to solve Problem (OP1) using SciPy's implementation of SLSQP [27]. Due to acquiring good results early with this method, no other solver or technique was tested. If the methods proposed in this study were to be refined and optimized, many more solvers and objective functions would have been tested, as discussed further in Section 5.3.

3.3.3 Adjusting the Translational Prediction

An issue with the local prediction, $\hat{\boldsymbol{d}}$, made in Section 3.3.2, is that it relies solely on local and past information. To address this limitation, the local prediction was treated as a tentative prediction, serving as a guide to identify the space where a more suitable prediction might lie. To achieve this, the following strategy was employed: within an angle α around the direction $\hat{\boldsymbol{d}}$, a spherical Fibonacci mapping generated n new candidate directions [26], illustrated in Figure 3.5. The parameter values used were $\alpha = 25°$ and $n = 50$.

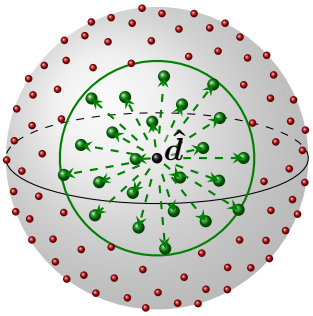


Figure 3.5: Search directions generated around the initial predicted translation $\hat{d}$.

A line search was performed in each candidate direction, where a maximum clearance was found on each line. The candidate direction corresponding to the largest clearance was then chosen as the new predicted translation direction $\boldsymbol{d}$, replacing $\hat{\boldsymbol{d}}$. This process is illustrated in Figure 3.6.

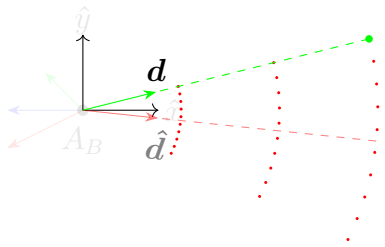


Figure 3.6: Search directions in the readjustment phase of predicting the translation. Red dots represent search directions, the green dot illustrates the point where the rigid body had the greatest clearance.

3.3.4 Aligning Anchor Frame with Predicted Translation

Once a final prediction had been made about the translation direction for the subsequent step, the anchor frame was rotated to align the discretization of the path planner with the predicted translation, as shown in Figure 3.7. This alignment aims to capitalize on the likelihood of a more open configuration space in the direction of translation, facilitating easier navigation, as discussed in Section 1.3, and visually depicted in Figure 1.10b.

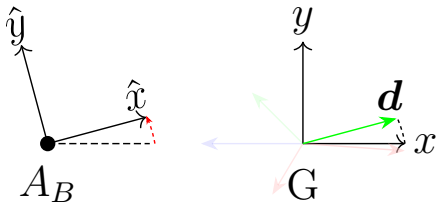


Figure 3.7: Illustration of a rigid body's anchor frame, A_B, being rotated to align with the globally predicted translation, $\boldsymbol{d}$.

3.3.5 Summary of the Orientation Method

The integration of the anchor frame orientation strategy with the path planning algorithm is a critical aspect of this study, aiming to increase the navigability of the configuration space for path planning. The process involves a series of well-defined steps, as described in the pseudocode of Algorithm 2 and further explained in the previous subsections.

Initially, before the first step of the path planning algorithm, measurements are collected by inducing small displacements to the rigid body. For each new configuration, the points on the rigid body and static geometry that create the smallest distance between them are recorded. These measurements, referred to as collision vectors, are used in an optimization step to predict the translation direction for

the next step. This prediction, denoted by $\hat{\boldsymbol{d}}$, is preliminary. Specifically, the optimization step minimizes the sum of the scalar products between $\hat{\boldsymbol{d}}$ and all collision vectors, essentially finding a direction that deviates as much as possible from the directions of the collision vectors.

Once the prediction $\hat{\boldsymbol{d}}$ is established, a search is conducted within a cone centered around $\hat{\boldsymbol{d}}$. This search involves temporarily translating the rigid body along various directions evenly distributed within the cone. The direction yielding the highest clearance for the rigid body is then selected as the new predicted direction, $\boldsymbol{d}$. The alignment of the anchor frame follows, where one of its axes is aligned with $\boldsymbol{d}$, thus optimizing the path planner's discretization to potentially enhance the navigability of the configuration space in the subsequent step.

Find New Orientation of Anchor Frame
Purpose: Predict the coming translational movement and align one of the axes of the anchor frame with it
Procedure:

1. **IF** it is the first step **THEN**
   Generate measurements
   ELSE
   Get measurements made by the path planner in previous step
2. Calculate collision from measurements in current pose
3. Optimize over collision vectors to get local translation prediction, $\hat{d}$.
4. Around $\hat{d}$, generate evenly distributed search directions in a cone
5. **FOR EACH** search direction **DO**
   (a) Translate rigid body along search direction, stopping at collision
   (b) Measure clearance in all poses
   (c) Keep track of search direction with the highest clearance
   END
6. Align one of the axes of the anchor frame with the search direction, d, associated with the highest clearance

Algorithm 2: Applying the orientation strategy to the anchor frame.

3.4 Anchor Frame Placement Strategy

The following methodology is based on the hypothesis that positioning the rigid body's anchor frame to increase rotational freedom will yield a more navigable configuration space. The strategy is summarized in two main steps:

1. **Gain insights on the forthcoming motion/twist**: Use available data to predict the next step's transformation. The prediction is in the form of a twist.

2. **Place the anchor frame on the predicted screw axis**: Based on the predicted twist, find the corresponding screw axis and place the anchor frame there, in order to facilitate easier navigation through the configuration space.

In contrast to the development of the orientation method presented in Section 3.3.1, this method does not use any preliminary verification, such as a preview step, to confirm the existence of a good placement method, as no effective analogy to the displacement vector was found. Initially, the full transformation of the step was tested using a logarithmic approach to obtain the corresponding twist. This immediately resulted in extremely long runtimes, possibly due to the non-surjective nature of the matrix logarithm, and thus this preview step was dismissed.

Without the above-mentioned verification, a model was developed to predict a screw axis that yields a high rotational freedom. The data used were the collision vectors from Section 3.1.2, and when no collision vectors were available, artificial initial measurements were made according to Section 3.3.1.

3.4.1 Local Twist Prediction

Just as in Section 3.3.1, the coming movement of the rigid body is to be predicted. However, instead of limiting the prediction to translational degrees of freedom, the twist that is to act on the rigid body was predicted.

Consider the function $\gamma : \mathbb{R}^6 \mapsto SE(3)$, which is parameterized by the twist $\boldsymbol{\xi}$, that outputs the corresponding rigid body transformation,

$$\gamma(\boldsymbol{\xi}) = \exp\left(\bar{\xi}\right), \tag{3.2}$$

where $\bar{\xi}$ is the homogeneous representation of the twist. Note that $\gamma(\mathbf{0})$ is the identity transformation.

A similar scenario to that in Figure 3.1, is depicted in Figure 3.8, but for general transformations rather than pure translations. The collision vector $\boldsymbol{c}$, pointing from $\boldsymbol{p}$ to a point $\boldsymbol{q}$ on the static geometry, is used in this prediction as well. It is accompanied by the vector $\boldsymbol{b}$ which points from the anchor frame, A_B, to $\boldsymbol{p}$. Since any rotation of the rigid body B is defined about A_B, the anchor frame affects the predicted twist.
One measure of how $\boldsymbol{\xi}$ moves $\boldsymbol{p}$ relative to $\boldsymbol{q}$ (and thus helps avoid collision) is given by

$$h_{(\boldsymbol{p},\boldsymbol{q})}(\boldsymbol{\xi}) = (\gamma(\boldsymbol{\xi})\bar{\boldsymbol{p}} - \gamma(\mathbf{0})\bar{\boldsymbol{p}}) \cdot \bar{\boldsymbol{c}}. \tag{3.3}$$

Note that the expression in Equation (3.3) is precisely $\boldsymbol{a} \cdot \boldsymbol{c}$ in Figure 3.8.

The gradient of the function in the "current" configuration ($\boldsymbol{\xi} = \mathbf{0}$) has the form

$$\nabla h_{(\boldsymbol{p},\boldsymbol{q})}(\mathbf{0}) = \begin{bmatrix} \boldsymbol{b} \times \boldsymbol{c} \\ \boldsymbol{c} \end{bmatrix}, \tag{3.4}$$

which is easily computable from the vectors $\boldsymbol{b} = \boldsymbol{p} - \mathbf{0}$ and $\boldsymbol{c} = \boldsymbol{q} - \boldsymbol{p}$.

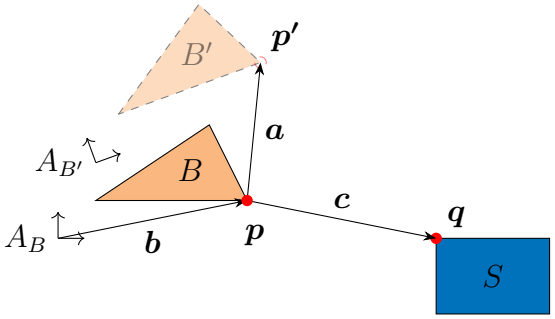


Figure 3.8: A twist $\boldsymbol{\xi}$ defines the transformation $\gamma(\boldsymbol{\xi})$ that acts on the rigid body, B. The point $\boldsymbol{p'}$ is where $\boldsymbol{p}$ ends up after the transformation, given in homogeneous coordinates by $\bar{\boldsymbol{p'}} = \gamma(\boldsymbol{\xi})\bar{\boldsymbol{p}}$

A derivation of the expression in (3.4) is given in Appendix A.1.1. For a set of collision points $\{(\boldsymbol{p}_j, \boldsymbol{q}_j)\}_{j=1}^N$, the predicted twist is computed as

$$\begin{aligned} \underset{\boldsymbol{\xi}=\left[\begin{smallmatrix}\boldsymbol{\omega}\\ \boldsymbol{v}\end{smallmatrix}\right]}{\text{minimize}} \quad & \sum_{j=1}^{N} \frac{\boldsymbol{\xi} \cdot \nabla h_{(\boldsymbol{p}_j, \boldsymbol{q}_j)}(\boldsymbol{0})}{\|\nabla h_{(\boldsymbol{p}_j, \boldsymbol{q}_j)}(\boldsymbol{0})\|} \\ \text{subject to} \quad & \frac{||\boldsymbol{\omega} \times \boldsymbol{v}||}{||\boldsymbol{\omega}||^2} \leq l, \\ & ||\boldsymbol{\xi}|| = 1, \end{aligned} \tag{OP4}$$

where l represents a distance. The optimization problem aims to find the twist $\boldsymbol{\xi}$ that minimizes the objective function in (OP4), with the goal of avoiding collisions. This objective is a natural generalization of the objective function in Problem (OP1), when generalizing from translations to twists. The constraint $\frac{||\boldsymbol{\omega}\times\boldsymbol{v}||}{||\boldsymbol{\omega}||^2} \leq l$ ensures that the movement of the anchor frame, defined in Chapter 2.4.2, remains close to the physical bounds of the rigid body. Specifically, l was chosen based on a smaller series of simulation results, which indicated that setting l to one fourth of the shortest side of the bounding box consistently led to fewer smallest distance computations compared to both longer and shorter values of l. The normalization constraint $||\boldsymbol{\xi}|| = 1$ importantly prohibits the vanishing of twists. It is included because some sort of vicinity constraint is needed; controlling the rigid body from a very distant anchor frame is like eating food with an excessively long fork, in that the precision is lost. This behavior has been verified in the IPS software suite, in that the path planning algorithm struggles to find paths when the anchor frame is far away from the rigid body.

3.4.2 Perturbation Analysis

After predicting a twist $\boldsymbol{\xi}^{(0)}$, a perturbation analysis is performed by introducing noise. This involves the following steps:

1. **Generating Noise Vectors:**
   - Noise vectors $\boldsymbol{\varepsilon}^{(j)} \in \mathbb{R}^6$ are generated for $j = 1, \ldots, 20$, where each $\boldsymbol{\varepsilon}^{(j)} \sim \mathcal{N}(\boldsymbol{0}, \Sigma)$. The choice to use 20 noise vectors is based on heuristic

reasoning; adding more than 10 noise vectors has a minimal impact on the number of smallest distance computations, thus doubling the amount provides a reasonable margin.
- The covariance matrix Σ is diagonal: $\Sigma = \text{diag}(\sigma_{\omega_1}, \sigma_{\omega_2}, \sigma_{\omega_3}, \sigma_{v_1}, \sigma_{v_2}, \sigma_{v_3})$.
- Each $\sigma_\alpha = 0.10|\alpha|$ for $\alpha \in \{\omega_1, \omega_2, \omega_3, v_1, v_2, v_3\}$, representing 10 percent of the magnitudes of the corresponding components of $\boldsymbol{\xi}^{(0)}$. The choice of 10 percent is made heuristically, as it yields perturbations in the IPS software suite that look small yet clearly visible.

2. **Generating Perturbed Predictions:**
   - Each noise vector $\boldsymbol{\varepsilon}^{(j)}$ is then added to $\boldsymbol{\xi}^{(0)}$ to generate candidate twists as:
   $$\boldsymbol{\xi}^{(j)} = \boldsymbol{\xi}^{(0)} + \boldsymbol{\varepsilon}^{(j)}, \quad i \in \{1, \ldots, 20\}.$$
3. **Expanding Candidate Twists:**
   - The set of candidate twists is expanded by including the zero twist (no change) and the original prediction:
   $$\mathcal{S}_{\boldsymbol{\xi}} = \{\mathbf{0}, \boldsymbol{\xi}^{(0)}, \boldsymbol{\xi}^{(1)}, \ldots, \boldsymbol{\xi}^{(20)}\}.$$
4. **Moving the Anchor Frame:**
   - The anchor frame is moved only if a sufficient number of twist candidates in $\mathcal{S}_{\boldsymbol{\xi}} \setminus \{\mathbf{0}\}$ increase the rotational freedom compared to $\mathbf{0}$. The threshold for "sufficiently" many is determined to be 30%, meaning that at least seven out of the 21 candidates in $\mathcal{S}_{\boldsymbol{\xi}} \setminus \{\mathbf{0}\}$ need to outperform $\mathbf{0}$. This value is chosen based on preliminary experiments showing that a 30% threshold provides a balance between computational efficiency and accuracy in predicting beneficial twists.
   - The anchor frame is moved to the screw axis given by the candidate twist that provides the greatest rotational freedom. This freedom is defined by the maximum collision-free rotation, either clockwise or counterclockwise, around the corresponding screw axis.

3.4.3 Summary of the Placement Strategy

The algorithm presented in this section is a preliminary version aimed at exploring the efficiency impact of strategically placing the anchor frame. To isolate the effect of anchor frame placement, no alignment adjustments are made in this algorithm. In Algorithm 3, the reader finds pseudocode for this algorithm. The outline is the same as in Algorithm 1. Importantly, each step begins with the anchor frame's position being restored relative to the geometry of the rigid body. This action, combined with the first constraint of Problem (OP4), assures that the anchor frame does not stray too far from the rigid body.

Find New Placement of Anchor Frame

Purpose: Find an axis that increases rotational freedom. Move the anchor frame to that axis.

Procedure:

1. Assure the anchor frame is in its default position with respect to the rigid body, that is, centered in a bounding sphere.
2. **IF** it is the first step **THEN**
   Generate initial measurements
   ELSE
   Get measurements made by the path planner in previous step
3. Calculate collision from measurements in current pose
4. Optimize over collision vectors to get local twist prediction, $\boldsymbol{\xi}$.
5. Around $\boldsymbol{\xi}$, generate candidate twists from Gaussian distribution and include the zero vector.
6. **FOR EACH** candidate twist **DO**
   (a) Calculate the corresponding placement of the anchor frame, such that it lies on the anchor frame
   (b) Record the rotational freedom about the screw axis
   END
   IF sufficiently many candidate twists increase rotational freedom **THEN**
   Place the anchor frame according to the twist associated with the highest clearance
   ELSE
   Leave the anchor frame in its default position

Algorithm 3: Applying the placement strategy to the anchor frame.

4
Tests and Results

In this chapter, the key findings from the exploration phase and strategy evaluation are outlined. As noted in previous chapters and further demonstrated in Section 4.1.1, the efficiency of the path planning algorithm depends significantly on the initial orientation of the anchor frame's pose. Users of the path planning algorithm typically do not concern themselves with the orientation of the anchor frame, neither should they, as it would make the user experience more complicated. This means that one can expect arbitrary initial orientations in practice. Therefore, performance is measured by comparing the distribution of the smallest distance computations for evenly distributed initial orientations between a placement strategy, an orientation strategy, and non-action. While a robust strategy may sometimes underperform compared to taking no action at all for certain orientations, it generally performs better on average, across all initial orientations. All evenly distributed initial orientations were generated using the method described in Section 3.1.3.

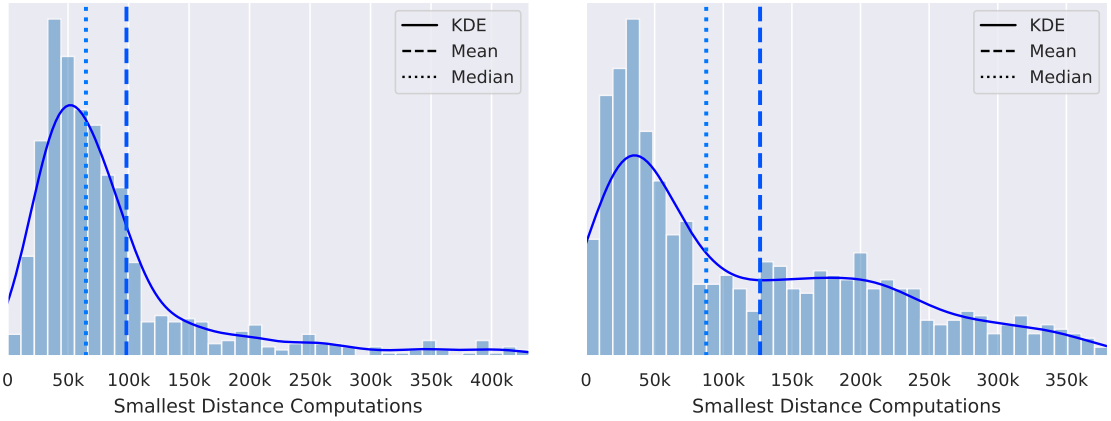
4.1 Exploration

This section examines the overall impact that placement and orientation have on the path planning algorithm. Except for setting the initial orientation, no action is taken and stepwise path planning, as discussed in Section 3.1.1, is run. The results from this section are used to assess the performance of the placement and orientation strategies.

4.1.1 Orientation

The performance impact of evenly distributed initial orientations for the two break booster scenes is shown in Figure 4.1. Each histogram is supplemented by a Kernel Density Estimation (KDE) line [28], which provide a smooth density function for easier comparisons.

In addition to the histograms displaying the overall performance distribution, it is also of interest to determine if certain directions yield stronger results than others. Figure 4.2 shows the performance distribution over all initial orientations on the unit sphere. Each vertex of the sphere is one of the 256 evenly distributed points on the unit sphere which an arbitrary axis of the rigid body's anchor frame is pointed towards, as described in Section 3.1.3. The colors of the triangles' faces represent the mean performances over their vertices, including the two rotations around each



(a) Performance distribution over evenly distributed orientations for the *Brake Booster 1* scene.

(b) Performance distribution over evenly distributed orientations for the *Brake Booster 2* scene.

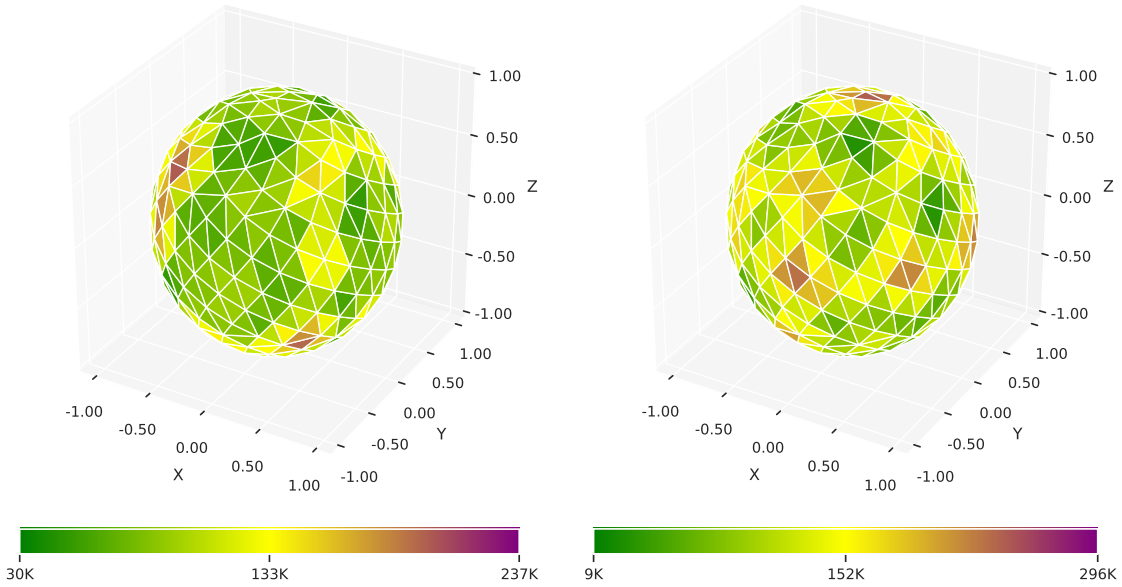
Figure 4.1: Histograms over performance under evenly distributed initial orientations for two similar scenes, *Brake Booster 1* and *Brake Booster 2*.

point by 120°.

Table 4.1 summarizes key statistics for the performance distributions shown in Figures 4.1 and 4.2, together with the results for the remaining scenes.

Table 4.1: Performance statistics in terms of number of smallest distance computations for calculating a full path, without employing an anchor frame strategy. Run over evenly distributed initial orientations.

Scene	Min	Median	Mean	Max
Brake Booster 1	6 796	64 790	98 285	502 147
Brake Booster 2	5 221	87 584	126 824	660 953
Center Console	9 270	33 774	78 098	1 634 917
Motorcycle Carburetor	15 373	73 204	86 061	320 939
Seatbelt Retractor	6 907	22 013	22 356	73 091



(a) Performance distribution over evenly distributed orientations for the *Brake Booster 1* scene.

(b) Performance distribution over evenly distributed orientations for the *Brake Booster 2* scene.

Figure 4.2: Spheres made up of a triangle mesh, built from the points on the sphere, towards each of which an axis of the anchor frame is pointed, generated from the Fibonacci Spherical Mapping. The faces of the triangles represent the mean performance for all vertices that make up the triangle. The spheres' backsides cannot be seen, but follows the same general noisy patterns as in the front.

4.1.2 Placement

The performance impact of anchor frame placement is exemplified in Figure 4.3, in a 3D lattice. The figures represent two runs for the exact same path planning problem where the only difference is the initial orientation of the scene. The scene in this example is *Seatbelt Retractor* and is depicted in Figure 4.4. No actions were taken during the stepwise path planning once the anchor frame was set. The values are expressed relative to a reference value, which is the number of smallest distance computations that were made for the default anchor frame position for each initial orientation. Note that the relative performance gain from moving the anchor frame is dependent on the performance of the default anchor frame for that orientation. Clearly, there are better and worse areas.

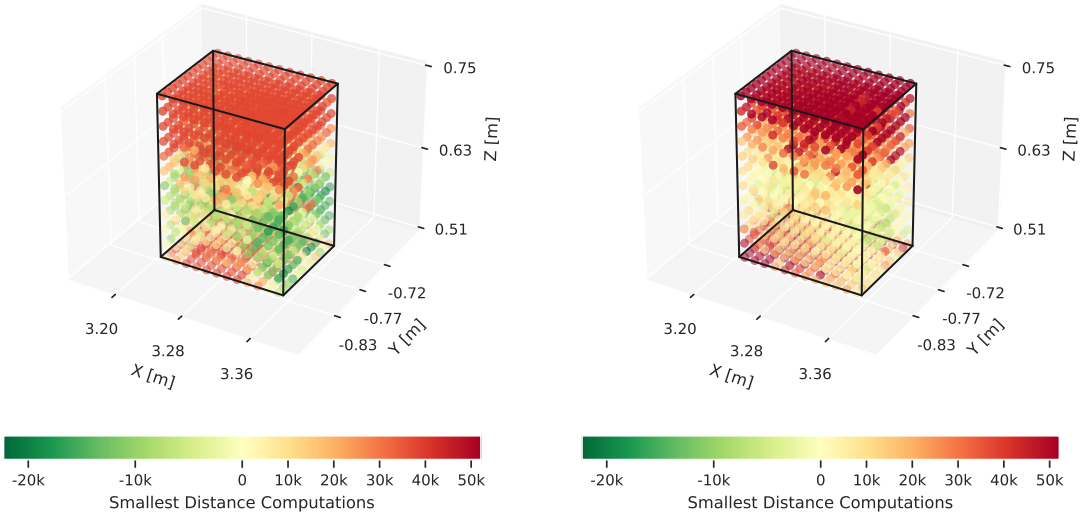


Figure 4.3: Performance distributions over anchor frame positions in a cubic lattice within the rigid body bounding box, for two different initial orientations in the *Seatbelt Retractor* scene. The values represent the difference between the total number of smallest distance computations for each moved anchor frame and the reference value, which equals the number of smallest distance computations in the default position for each orientation. Each run was aborted immediately if the number of smallest distance computations surpassed an upper limit.

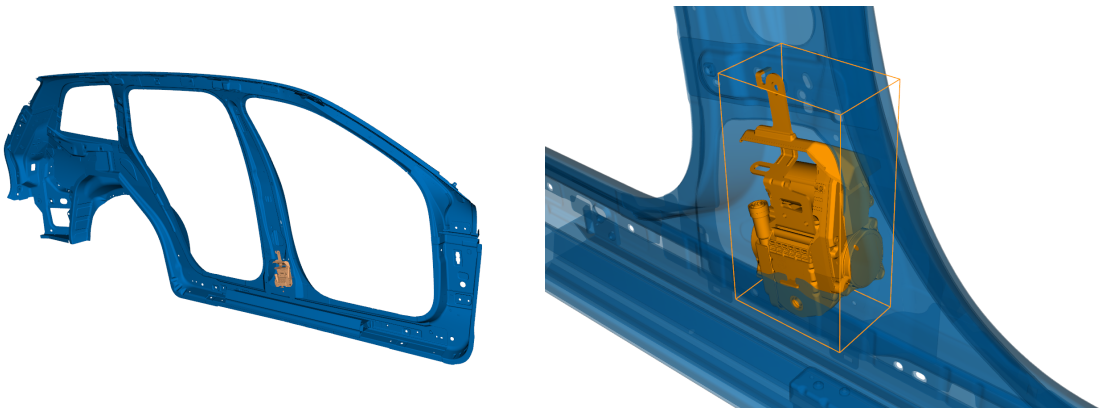


Figure 4.4: The *Seatbelt Retractor* scene in its entirety and a close up of the challenging parts of the path planning problem. The rigid body's bounding box is also displayed. Model by courtesy of Volvo Cars.

Using the same data, Figure 4.5 shows the distribution of differences between the reference value and performance under various anchor frame placements. The characteristics of these histograms are quite different from those of Figure 4.1, in that the distribution looks more elongated. While keeping in mind that different geometries are used, the elongated histograms imply that there is no particular tendency towards small performance changes when moving the anchor frame. Furthermore, the medians of the swept anchor frame are in both cases higher than the reference median, indicating an overall performance decrease when randomly moving the anchor frame.

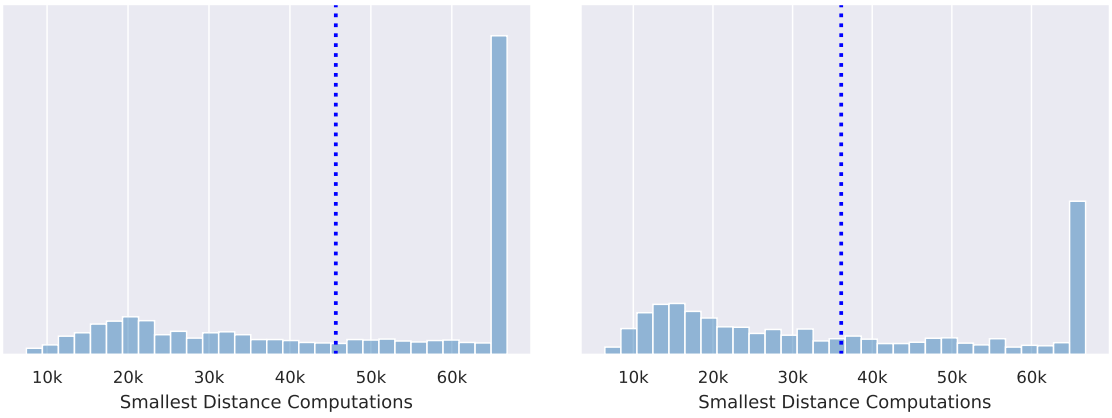


Figure 4.5: Performance distributions over anchor frame positions in a cubic lattice within the rigid body bounding box, for two different initial orientations. The values represent the difference between the total number of smallest distance computations for each moved anchor frame and the reference value, which is the number of smallest distance computations in the default position for each orientation. A run was aborted once the number of smallest distance computations had surpassed an upper limit.

The *Brake Booster 2* scene, which may not be shown due to confidentially, does not have as promising results, as illustrated in Figure 4.6. There is some indication of areas which are generally better or worse, but there is clearly a lot of noise. As mentioned in Section 3.2.2, running all $12^3 = 1728$ instances for just one initial orientation could take roughly 24 hours of computing time, so all scenes have not been tested within this framework. Ideally, all five scenes would be tested by computing the mean over all 768 evenly distributed initial orientations for each anchor frame placement, but running that simulation could take up to ten years of computing time.

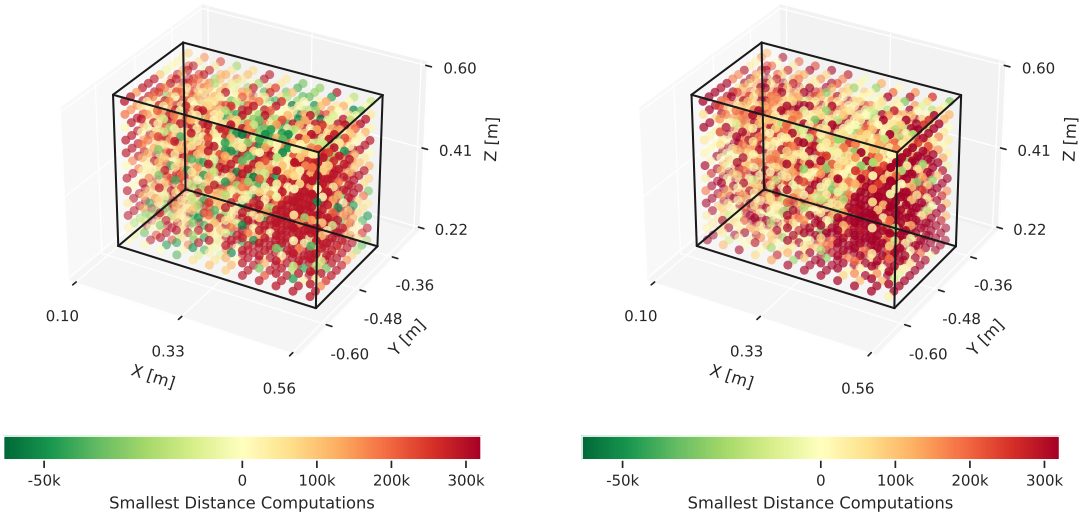


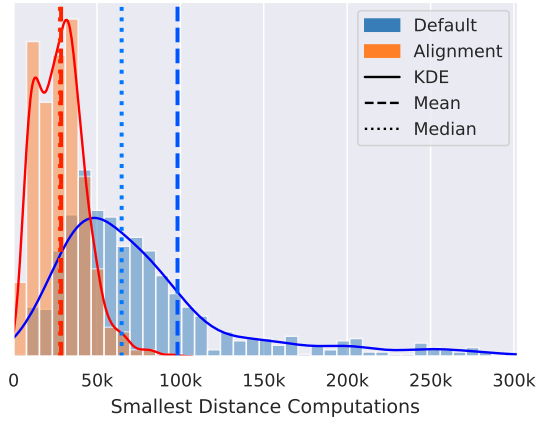
Figure 4.6: Performance differences histogram for the *Brake Booster 2* scene over various anchor frame positions, for two different initial orientations. The values represent the difference between the smallest distance computations for each moved anchor frame and the reference value, which is the number of smallest distance computations in the default position for each orientation. A run was aborted once the number of smallest distance computations had surpassed an upper limit.

4.2 Strategies

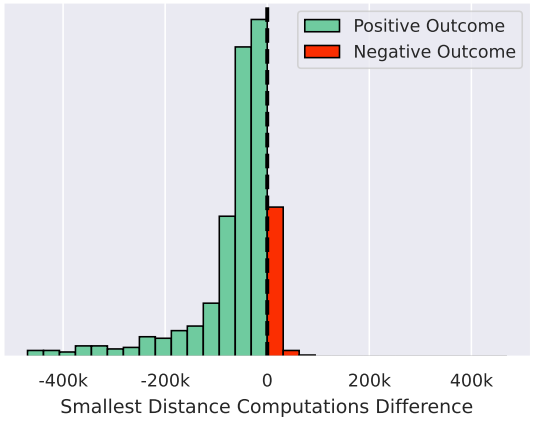
In this section, the strategies formulated in Chapter 3 will be compared against the performance metrics acquired in the previous section. Each histogram displays performance data for one scene, and comes in two types. A comparison histogram shows the performance of the strategy in orange bins, with the default performance in blue bins. A difference histogram shows the difference between the number of smallest distance computations per initial orientation, providing insight into how many orientations benefited from the strategy. A red bin in this histogram indicates a decrease in performance, whereas a green bin indicates an improvement.

4.2.1 Orientation

The results for the alignment strategy for the *Brake Booster 1* scene, depicted in Section 1.4, are shown in Figure 4.7. The alignment strategy, represented by the orange histogram, is in Figure 4.7a compared against the default distribution. Individual runs and their differences are featured in Figure 4.7b. Similar results, this time for *Brake Booster 2*, are presented in Figure 4.8.

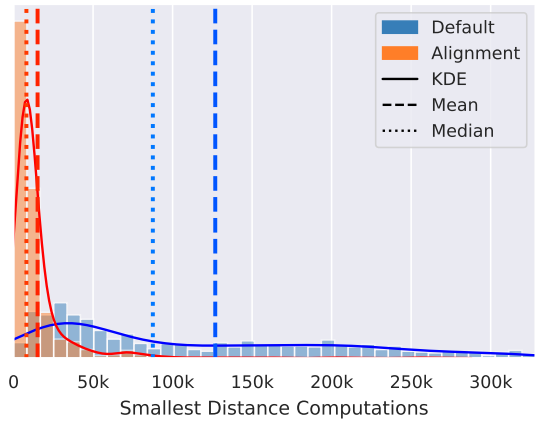


(a) Overlapping histograms over evenly distributed orientations with and without alignment.

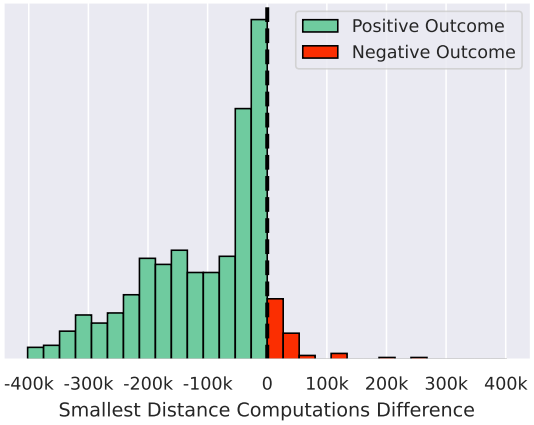


(b) Differences between individual runs with identical initial orientations with and without alignment.

Figure 4.7: Comparison between the performance of the *Brake Booster 1* scene with and without alignment.



(a) Overlapping histograms over evenly distributed orientations with and without alignment.

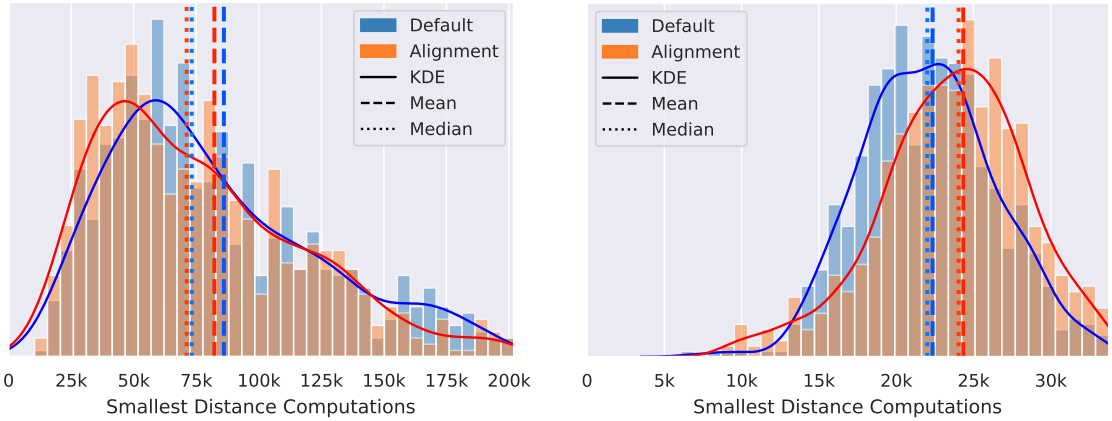


(b) Differences between individual runs with identical initial orientations with and without alignment.

Figure 4.8: Comparison between the performance of the *Brake Booster 2* scene with and without alignment.

Two additional scenes, where the effects of the alignment strategy is not as prevalent, are displayed in Figure 4.9.

As for the impact analysis, key statistics for the performance of the orientation strategy are highlighted in Table 4.2.



(a) Overlapping histograms over evenly distributed orientations with and without alignment for the *Seatbelt Retractor* scene.

(b) Overlapping histograms over evenly distributed orientations with and without alignment for the *Center Console* scene.

Figure 4.9: Performance for two scenes that are not much affected by alignment.

Table 4.2: Statistics for the performance of the path planning algorithm when employing the orientation strategy. The performance metric is again the number of smallest distance computations, and it is measured over evenly distributed initial orientations.

Scene	Min	Median	Mean	Max
Brake Booster 1	3 921	27 771	28 486	95 292
Brake Booster 2	4 569	8 082	15 107	269 205
Center Console	7 485	34 376	74 605	550 934
Motorcycle Carburetor	11 502	71 168	82 263	348 643
Seatbelt Retractor	8 964	24 036	24 337	86 252

The differences between Table 4.2 and Table 4.1 can be seen in Table 4.3. Bold numbers represent measurements which benefited from using the alignment strategy for a certain scene. Note that the crucial stats are the median and mean values for the scenes. The minimum and maximum values are not quite as important, as the path planning algorithm can easily get stuck in or completely avoid challenging parts of the configuration space. Such cases are most likely impossible to predict.

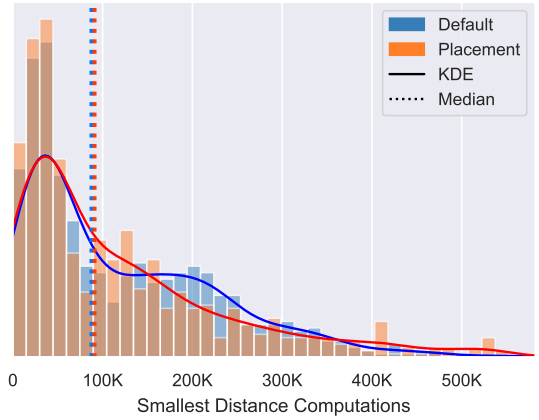
Table 4.3: Differences of metrics between the alignment method and the default method. Negative values correspond to the alignment strategy performing better than the default strategy. Metrics considered are minimum, median, mean, and maximum number of smallest distance computations.

Scene	Min	Median	Mean	Max
Brake Booster 1	**-2 876**	**-37 019**	**-69 798**	**-406 855**
Brake Booster 2	**-652**	**-79 502**	**-111 716**	**-391 748**
Center Console	**-1 785**	601	**-3 493**	**-1 083 983**
Motorcycle Carburetor	**-3 871**	**-2 036**	**-3 797**	27 704
Seatbelt Retractor	2 057	2 022	1 981	13 161

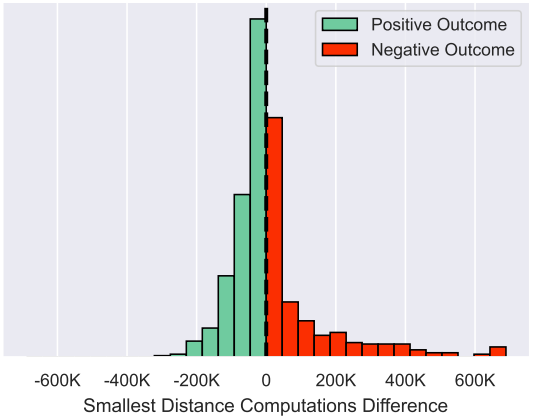
4.2.2 Placement

The results for the placement strategy are presented slightly differently from the results in Section 4.2.1. Most notably, this strategy performed worse than both the alignment strategy and the default strategy. Additionally, relocating the anchor frame may cause excessively high runtimes, making simulations impractically long. To address this, runs were interrupted once they reached an upper limit set relative to the median runtime of the default strategy. These attempts are referred to as "failed", and the interruption thresholds were set to eight times the median value for the default strategy. Consequently, the mean and maximum number of smallest distance computations lose their meaning; interruption thresholds yield the worst values, whereas values from interrupted runs should not contribute to a mean value. The median values presented account for both the successful and the failed runs. One additional important statistic is the so-called "usage". This refers to the relative frequency of more than 30% of the candidate twists increasing rotational freedom, which results in the placement strategy being used, as described in Section 3.4.2.

The results for the *Brake Booster 2* scene are presented in Figure 4.10. Contrary to the orientation strategy, the placement strategy has little effect on performance. For scenes where the orientation strategy does not improve the performance, it at least produces results very similar to the default strategy. The placement strategy, however, seems to generate long-tailed distributions for most scenes. Another example of this is given in Figure 4.11, which gives a performance comparison for the *Seatbelt Retractor* scene.

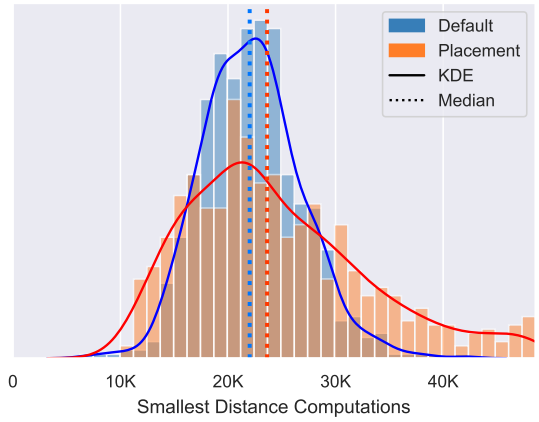


(a) Overlapping histograms over evenly distributed orientations with and without placement.

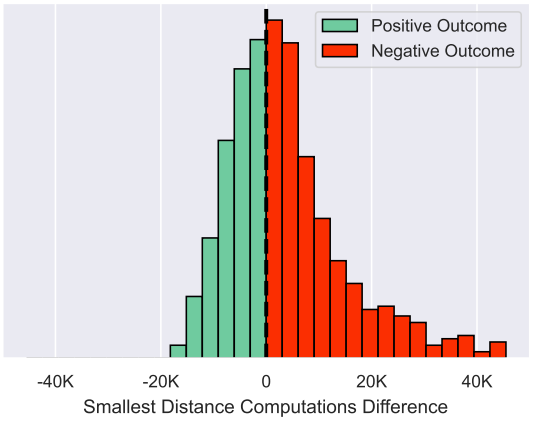


(b) Differences between individual runs with identical initial orientations with and without placement.

Figure 4.10: Comparison between the performance of the *Brake Booster 2* scene with and without placement of the anchor frame.



(a) The somewhat symmetric and concentrated distribution of the default method is contrasted by the long-tailed placement performance.



(b) Difference histogram comparing the placement strategy with the default strategy for the *Seatbelt Retractor* scene.

Figure 4.11: The placement strategy has negative impact on the performance for the *Seatbelt Retractor* scene.

The scene which yielded the worst performance out of the five scenes was *Brake Booster 1*; see Figure 4.12. Roughly a quarter of all runs had to be interrupted after reaching the limit of eight times the reference value.

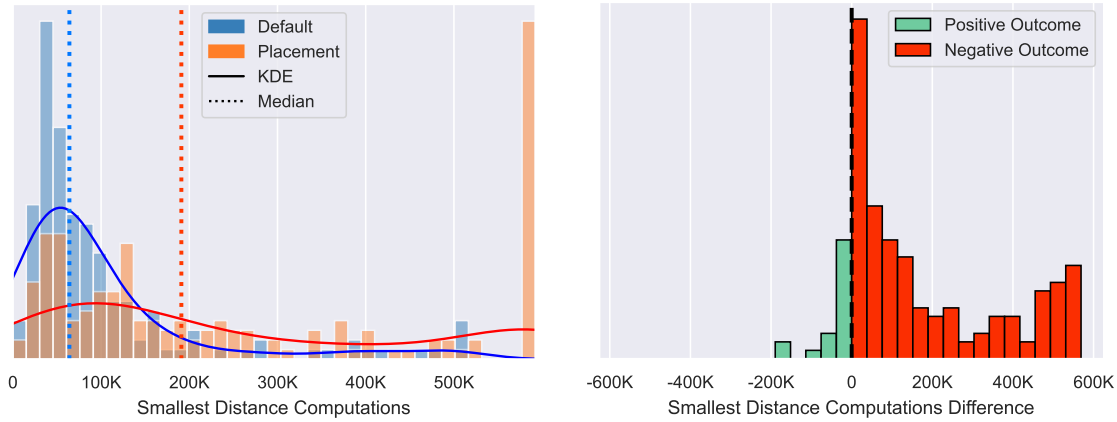


Figure 4.12: The placement strategy has a dramatically negative impact on the performance of the path planning algorithm for the *Brake Booster 1* scene.

Key statistics are highlighted in Table 4.4. Note that the maximum number of smallest distance computations, and the mean number of smallest distance computations, have been removed. In their place, the failure rate and the usage rate are shown. For the minumum values and median values, the differences between the placement strategy and the default strategy (see Table 4.1) are also listed. The difference columns are highlighted by Δ_{Min} and Δ_{Median}. All of their values are positive, meaning that both the median values and the minimum values were worsened across all scenes.

Table 4.4: Performance statistics for the path planning algorithm when employing the placement strategy. Run over evenly distributed initial orientations.

Scene	Min	Δ_{Min}	Median	Δ_{Median}	Failure	Usage
Brake Booster 1	12 477	5 681	191 007	126 217	26.5%	67.3%
Brake Booster 2	6 129	908	90 980	3 396	4.17%	41.0%
Center Console	15 012	5 742	50 945	17 171	9.38%	79.9%
Motorcycle Carburetor	16 329	956	127 430	54 226	5.21%	51.1%
Seatbelt Retractor	10 004	3 097	23 631	1 618	0.13%	60.8%

5
Discussion

This chapter interprets the findings from the previous sections, providing a comprehensive analysis of the results. It examines the impact of anchor frame orientation and placement on the path planning algorithm's efficiency, evaluates the efficiency of the implemented strategies, and identifies encountered limitations. Additionally, it outlines potential areas for future research and concludes with key insights from the study.

5.1 Anchor Frame Pose Impact and Strategies

This section discusses the impact of anchor frame orientation and placement, as well as the developed strategies.

5.1.1 Orientation

The performance of the path planning algorithm is highly dependent on the orientation of the anchor frame. As shown in Table 4.1, the difference in the number of smallest distance computations can range from a few thousand to over a million for a single scene (*Center Console*). However, the spherical figures in Figure 4.2 suggest that determining a universally suitable orientation is challenging.

The alignment strategy works well for scenes where the main difficulty is purely translational. An early experiment involved planning one step ahead, reverting the step, and using the information from the planned step to predict the translation. This method's encouraging results led to the development of a strategy to approximate this approach using collision vectors. The final method was determined through trial and error.

The adjustment method was discovered somewhat by chance. The idea was to search in a cone based on the local prediction and find the maximum clearance along a search direction. Then, an additional optimization would occur using both old and newly acquired data. This approach initially failed, but while testing the code, the method of adjusting the prediction to point towards the maximum clearance was found to perform well.

A downside of the anchor frame orientation strategy is that there is no option to opt out of alignment. If no adjustment is made, it implies that the current orientation

is already aligned with the predicted translation. It might be possible to determine that alignment is not crucial if no significant translation is expected, but no such method was developed in this project.

5.1.2 Placement

The placement of the anchor frame significantly impacts the performance of the path planning algorithm. While a well-placed anchor frame offers benefits similar to a well-predicted translation, a poor placement can severely degrade the performance. Therefore, placing the anchor frame is more delicate than orienting it. In many scenes, the number of beneficial anchor frame positions is vastly outnumbered by those that harm performance. Unlike orientation, there is a default position that is generally a good option. Figure 4.3 shows that the performance landscape also depends on the initial orientation.

The placement strategy developed in this project has not seen great success. No strategy was found that uses runtime-available information to confirm whether a good strategy exists. One suspicion is that the predictions are very local. To correctly predict a larger rotation around an axis could require more data than what is available in the collision vectors. As seen when developing the orientation strategy, predicting the translation using only local and past information was insufficient; an additional cone search was necessary. The cone search's counterpart for the placement strategy was the perturbation analysis, which was implemented to avoid overfitting to local data. Interestingly, using the aggregated measure to decide whether to move the anchor frame performed much better than a method where the "best" candidate was chosen straight away.

Another part of the placement strategy that deserves further research is the optimization problem in Problem (OP4), in particular the objective function. It was chosen primarily because its three-dimensional counterpart in Problem (OP1) performed well in the orientation strategy. The reason for not researching the objective function more thoroughly was that development of the orientation strategy was prioritized, since that strategy was deemed more promising.

5.2 Challenges and Limitations

Moving the anchor frame is a fragile operation, more likely to cause harm than good without a proper strategy. For anchor frame orientation, a simple "cheat" confirmed the potential of alignment strategies. No such validation was found for anchor frame placement, thus more effort was spent on orientation strategies.

A challenging aspect of the project is controlling the state of an implicit, or unknown, six-dimensional configuration space. This space rarely behaves predictably and is difficult to make assumptions about. In typical use, the rigid body has to carefully wiggle and twist out, meaning there is quite little space to work with.

The length of a step, or how long the path planner runs before it automatically aborts, was not in our control. That meant for certain long steps, that there is a wide range of translational directions and rotations around several axes that can occur. It is likely not possible to strategically find one placement and orientation that works for all such steps.

5.3 Future Work

More work is needed on placement strategies for the anchor frame. Developing a successful verification, similar to the preview step for the orientation strategy, could help to identify where the current approach is failing, whether it's in using a screw axis, optimizing the twist, or collecting data. Once a successful verification is found, the next step is to replicate it using readily available data.

Another area not covered is the degree of freedom left after aligning the anchor frame's z-axis with the predicted translation. When the z-axis is pointed towards the predicted translation, the current method does not deal with how the other two axes should be oriented around that axis. This third, and last, rotational degree of freedom has shown to drastically affect performance, but no strategy has been found for dealing with it.

Currently, the length of a step cannot be adjusted. By allowing custom conditions to stop the path and realign the anchor frame, one could achieve more frequent alignment using better data to predict the next translation.

5.4 Conclusions

This study explored the impact of anchor frame orientation and placement on the performance of a path planning algorithm. The findings highlight the critical role of anchor frame orientation in improving algorithm efficiency. By predicting translations using local collision data and optimizing orientation towards areas of maximum clearance, computational efforts are significantly reduced. This approach underscores the potential for enhancing the path planner's efficiency performance through refined orientation strategies.

Conversely, anchor frame placement is more challenging. While well-placed anchor frames can improve the performance, a poor placement severely hinders it. The current placement strategy falls short due to the local nature of the available data and the difficulty in accurately predicting motions. This indicates the need for further research to develop robust placement strategies, possibly using more data.

The study also revealed several challenges and limitations. The six-dimensional configuration space and the inherent complexity of the path planning problem complicate the development of reliable strategies for both orientation and placement. Additionally, limited control over the path planner's step length and abort condi-

tions restricts the optimization efforts.

Investigating the remaining degree of freedom in anchor frame orientation could further enhance the performance of an already effective methodology. Even without additional adjustments, the orientation strategy appears to improve the overall efficiency of the path planning algorithm. We recommend IPS to test the current strategy across a wider variety of scenes and consider integrating it into their path planning algorithm. Additionally, defining custom abort conditions for the path planning algorithm could enhance prediction accuracy, supporting better alignment strategies. Future research on anchor frame strategies should also focus on developing effective placement strategies and methods for validating potential approaches.

In conclusion, while placement remains a complex challenge requiring further research, significant progress was made in optimizing the anchor frame orientation. The insights from this study can directly enhance path planning algorithms and serve as a foundation for future efforts to improve their efficiency and reliability.

Bibliography

[1] Fraunhofer Chalmers Centre. FCC Homepage. https://www.fcc.chalmers.se/, February 2024. Accessed on August 3, 2024.

[2] Fraunhofer. Fraunhofer Homepage. https://www.fraunhofer.de/en.html, July 2024. Accessed on July 18, 2024.

[3] Fraunhofer Chalmers Centre. FCC About Page. https://www.fcc.chalmers.se/about/, July 2024. Accessed on July 18, 2024.

[4] Industrial Path Solutions Sweden AB. Homepage of the IPS website. https://industrialpathsolutions.se/, 2024. Accessed on August 3, 2024.

[5] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[6] C. Zhou, B. Huang, and P. Fränti. A review of motion planning algorithms for intelligent robots. *Journal of Intelligent Manufacturing*, 33, 11 2021.

[7] W. Liu, Z. Huang, Y. Qu, and L. Chen. Path planning for robotic arms based on an improved RRT algorithm. *Open Journal of Applied Sciences*, 14:1214–1236, 2024.

[8] B. Feng, X. Jiang, B. Li, Q. Zhou, and Y. Bi. An adaptive multi-RRT approach for robot motion planning. *Expert Systems with Applications*, 252:124281, 2024.

[9] Y. Lu, Y. Ma, D. Hsu, and C. Pan. Neural randomized planning for whole body robot motion. *ArXiv*, abs/2405.11317, 2024.

[10] H. Huang, X. Wen, M. Niu, Md. S. Miah, T. Gao, and H. Wang. Multi-UAVs assisted path planning method for terrain-oriented air-ground collaborative vehicular network architecture. *IEEE Transactions on Intelligent Vehicles*, pages 1–11, 2024.

[11] J. Zaragoza Chichell, A. Rečková, M. Bizzarri, and M. Bartoň. Collision-free tool motion planning for 5-axis CNC machining with toroidal cutters. *Computer-Aided Design*, 173:103725, 2024.

[12] United Nations Environment Programme. Global resources outlook 2024: Bend the trend–pathways to a liveable planet as resource use spikes. https://wedocs.unep.org/20.500.11822/44901, 2024.

[13] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[14] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC press, 1994. PDF: https://www.cse.lehigh.edu/~trink/Courses/RoboticsII/reading/murray-li-sastry-94-complete.pdf.

[15] B. Graf. Quaternions and dynamics. *arXiv: Dynamical Systems*, 0811:2889, 2008.

[16] T. Lozano-Perez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, C-32:108–120, 1983.

[17] A. J. Hanson and H. Ma. Quaternion frame approach to streamline visualization. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):164–174, 1995.

[18] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[19] M. Farber, S. Tabachnikov, and S. Yuzvinsky. Topological robotics: motion planning in projective spaces. *International Mathematics Research Notices*, 2003.

[20] T.-W Kang, J.-G Kang, and J.-W Jung. A bidirectional interpolation method for post-processing in sampling-based robot path planning. *Sensors*, 21:7425, 2021. URL: `https://www.mdpi.com/1424-8220/21/21/7425`.

[21] D. Ferguson and A. Stentz. Using interpolation to improve path planning: The Field D* algorithm. *Journal of Field Robotics*, 23:79–101, 2006.

[22] Andreas Müller. Geometric interpolation of rigid body motions. In Masafumi Okada, editor, *Advances in Mechanism and Machine Science*, pages 310–320, Cham, 2023. Springer Nature Switzerland.

[23] M. Chasles. Note sur les propriétés générales du système de deux corps semblables entr'eux. *Bulletin de Sciences Mathématiques, Astronomiques Physiques et Chimiques*, 1830 (In French).

[24] G. Mozzi. *Discorso matematico sopra il rotamento momentaneo dei corpi*. Nella stamperia di D. Campo, 1763 (In Italian).

[25] M. Ceccarelli. Screw axis defined by Giulio Mozzi in 1763 and early studies on helicoidal motion. *Mechanism and Machine Theory*, 35(6):761–770, 2000.

[26] B. Keinert, M. Innmann, M. Sänger, and M. Stamminger. Spherical Fibonacci Mapping. *ACM Transactions on Graphics (TOG)*, 34:1–7, 2015.

[27] D. Kraft. A software package for sequential quadratic programming. Technical Report DFVLR-FB 88-28, DLR German Aerospace Center – Institute for Flight Mechanics, Köln, Germany, 1988.

[28] S. Węglarczyk. Kernel density estimation and its application. *ITM Web Conf.*, 23:00037, 2018. URL: `https://doi.org/10.1051/itmconf/20182300037`.

A

Mathematical Derivations

A.1 Twists Generate Rigid Body Transformations

This section shows how the expression in (2.14) is a rigid body transformation. Below, it will be assumed that any variable in $\mathbb{R}^3$ is time-dependent, parametrized by a time variable θ. Equation (2.10) can be written in matrix form,

$$\begin{bmatrix} \dot{\boldsymbol{p}}(\theta) \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{\bar{\omega}}(\theta) & \boldsymbol{v}(\theta) - \boldsymbol{\omega}(\theta) \times \boldsymbol{b}_0(\theta) \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{p}(\theta) \\ 1 \end{bmatrix}, \tag{A.1}$$

which includes the twist defined in Equation (2.11). Recalling the homogeneous representations of points from Chapter 2.2, we rewrite the differential equation as

$$\dot{\bar{\boldsymbol{p}}}(\theta) = \bar{\xi}(\theta)\bar{\boldsymbol{p}}(\theta), \tag{A.2}$$

which has the solution

$$\bar{\boldsymbol{p}}(\theta) = \exp\left(\int_0^\theta \bar{\xi}(\tau)\,\mathrm{d}\tau\right)\bar{\boldsymbol{p}}(0). \tag{A.3}$$

In particular, if $\bar{\xi}(\tau)$ is a constant $\bar{\xi}$ on the interval $\tau \in [0, \theta)$, then (A.3) simplifies to

$$\bar{\boldsymbol{p}}(\theta) = \exp(\bar{\xi}\theta)\bar{\boldsymbol{p}}(0), \tag{A.4}$$

which says, that a point on a rigid body (here $\boldsymbol{p}$) at a given point in time (here θ) is given by an exponentiated twist acting on the point's original position (i.e., $\boldsymbol{p}(0)$).

A.1.1 Deriving an Expression for Desirable Twists

This section motivates the gradients used in the anchor frame placement strategy. For a rigid body B and a static geometry S, consider a pair of points $(\boldsymbol{p}, \boldsymbol{q}) \in B \bigtimes S$. A twist in parameterized coordinates,

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{v} \end{bmatrix}, \tag{A.5}$$

is deemed desirable (at least locally) if it moves $\boldsymbol{p}$ further away from $\boldsymbol{q}$. Here, $\boldsymbol{\omega}$ denotes the angular velocity and $\boldsymbol{v}$ the translational velocity.
Assuming that γ represents the exponentiation of a twist, i.e.,

$$\gamma(\boldsymbol{\xi}) = \exp(\bar{\xi}), \tag{A.6}$$

define

$$h_{(\boldsymbol{p},\boldsymbol{q})}(\boldsymbol{\xi}) := (\gamma(\boldsymbol{\xi}) - \gamma(\mathbf{0}))\, \bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{c}}, \tag{A.7}$$

where $\bar{\boldsymbol{c}} = \bar{\boldsymbol{q}} - \bar{\boldsymbol{p}}$ is the homogeneous representation of the collision vector, and $h_{(\boldsymbol{p},\boldsymbol{q})}(\boldsymbol{\xi})$ is a measure of how much closer $\boldsymbol{\xi}$ moves $\boldsymbol{p}$ to $\boldsymbol{q}$. Since we specifically focus on the potential collision between $\boldsymbol{p}$ and $\boldsymbol{q}$ for the rest of the section, the indices are from now on dropped so that $h(\boldsymbol{\xi}) = h_{(\boldsymbol{p},\boldsymbol{q})}(\boldsymbol{\xi})$, and therefore $\nabla h = \nabla_{\boldsymbol{\xi}} h$ unless specified otherwise. Further, we define the vector $\boldsymbol{b} = \boldsymbol{p} - \mathbf{0}$. We have defined the position of the anchor frame as $\mathbf{0}$, for simplicity but without loss of generality since transformation matrices are designed for rotations around $\mathbf{0}$.

Claim 1: $\nabla h(\boldsymbol{\xi} = \mathbf{0}) = \begin{bmatrix} \boldsymbol{b} \times \boldsymbol{c} \\ \boldsymbol{c} \end{bmatrix}$.

In order to prove this, we first establish:

Claim 2: $\nabla h(\boldsymbol{\xi} = \mathbf{0}) = \nabla_{\boldsymbol{\xi}}((\bar{\xi}\bar{\boldsymbol{b}}) \cdot \bar{\boldsymbol{c}})$ for the case $\boldsymbol{\xi} = \mathbf{0}$.

Proof of Claim 2: By Taylor expansion about $\boldsymbol{\xi} = \mathbf{0}$,

$$h(\boldsymbol{\xi}) = \left(\left(\sum_{k=0}^{\infty} \frac{\bar{\xi}^k}{k!} - I \right) \bar{\boldsymbol{p}} \right) \cdot \bar{\boldsymbol{c}} = \left(\sum_{k=1}^{\infty} \frac{\bar{\xi}^k \bar{\boldsymbol{p}}}{k!} \right) \cdot \bar{\boldsymbol{c}} = \bar{\xi}\bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{c}} + \sum_{k=2}^{\infty} \frac{\bar{\xi}^k \bar{\boldsymbol{p}}}{k!} \cdot \bar{\boldsymbol{c}} \tag{A.8}$$

We show that all terms of second order or higher have zero gradients at $\boldsymbol{\xi} = \mathbf{0}$. Each term in the polynomial $\bar{\xi}^k \bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{q}}$ has at least two factors that are elements of $\boldsymbol{\xi}$, for all $k \geq 2$. This means that all terms in the elements of $\nabla \bar{\xi}^k \bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{q}}$ have at least one factor from $\boldsymbol{\xi}$, which means $\nabla \bar{\xi}^k \bar{\boldsymbol{p}} \cdot \bar{\boldsymbol{q}} = \mathbf{0}$ at $\boldsymbol{\xi} = \mathbf{0}$ for all $k \geq 2$. Thus,

$$\nabla h|_{\boldsymbol{\xi}=\mathbf{0}} = \nabla_{\boldsymbol{\xi}} \left(\bar{\xi}\bar{\boldsymbol{b}} \cdot \bar{\boldsymbol{c}} \right) + \underbrace{\nabla_{\boldsymbol{\xi}} \left(\sum_{k=2}^{\infty} \frac{\bar{\xi}^k \bar{\boldsymbol{b}}}{k!} \cdot \bar{\boldsymbol{c}} \right)}_{=0} = \nabla_{\boldsymbol{\xi}} \left(\bar{\xi}\bar{\boldsymbol{b}} \cdot \bar{\boldsymbol{c}} \right),$$

proving Claim 2.

Proof of Claim 1: We now compute $\nabla h(\boldsymbol{\xi} = \mathbf{0})$:

$$
\begin{aligned}
\nabla h(\boldsymbol{\xi} = \mathbf{0}) &= \nabla_{\boldsymbol{\xi}}((\bar{\xi}\bar{\boldsymbol{p}}) \cdot \bar{\boldsymbol{c}}) = \nabla_{\boldsymbol{\xi}}(\bar{\boldsymbol{p}}^T \bar{\xi}^T \bar{\boldsymbol{c}}) \\
&= \nabla_{\boldsymbol{\xi}} \left(\begin{bmatrix} p_x & p_y & p_z & 1 \end{bmatrix} \begin{bmatrix} \overset{\times}{\boldsymbol{\omega}} & \boldsymbol{v} \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} c_x \\ c_y \\ c_z \\ 0 \end{bmatrix} \right) \\
&= \nabla_{\boldsymbol{\xi}} \left(\begin{bmatrix} b_x & b_y & b_z & 1 \end{bmatrix} \begin{bmatrix} \overset{\times}{\boldsymbol{\omega}} & \boldsymbol{v} \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} c_x \\ c_y \\ c_z \\ 0 \end{bmatrix} \right) \\
&= \nabla_{\boldsymbol{\xi}} \left(\begin{bmatrix} b_x & b_y & b_z & 1 \end{bmatrix} \begin{bmatrix} \omega_z c_y - \omega_y c_z \\ -\omega_z c_x + \omega_x c_z \\ \omega_y c_x - \omega_x c_y \\ v_x c_x + v_y c_y + v_z c_z \end{bmatrix} \right) \\
&= \begin{bmatrix} \nabla_{\boldsymbol{\omega}} \\ \nabla_{\boldsymbol{v}} \end{bmatrix} (b_x(\omega_z c_y - \omega_y c_z) + b_y(-\omega_z c_x + \omega_x c_z) + b_z(\omega_y c_x - \omega_x c_y) \\
&\quad + v_x c_x + v_y c_y + v_z c_z) \\
&= \begin{bmatrix} b_y c_z - b_z c_y \\ b_z c_x - b_x c_z \\ b_x c_y - b_y c_x \\ c_x \\ c_y \\ c_z \end{bmatrix} = \begin{bmatrix} \boldsymbol{b} \times \boldsymbol{c} \\ \boldsymbol{c} \end{bmatrix},
\end{aligned}
$$

which proves Claim 1 and gives the expression used in the placement strategy. □

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY